Marine Puibaraud

Design of an automated system for the diagnosis of the Internet connection

School of Electrical Engineering

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 30.9.2013

Thesis supervisor:

Prof. Jukka Manner

Thesis advisor:

M.Sc. Sebastian Sonntag

Aalto University
School of Electrical
Engineering

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Marine Puibaraud

Title: Design of an automated system for the diagnosis of the Internet connection

Date: 30.9.2013 Language: English Number of pages:6+94

Department of Communications and Networking

Professorship: Networking Technology Code: S-38

Supervisor: Prof. Jukka Manner

Advisor: M.Sc. Sebastian Sonntag

The Internet has taken an important place in daily life, private and professional. It becomes more and more common for any company to run its business relying on Information and Communication Technologies services, whichever its size is. In this new era where technology trends tend to be real time-oriented, Internet connectivity is no more a plus, but a requirement. For any user for whom the Internet access is a necessity, being left out without Internet services can cause substantial damages to a business company for instance. Now users want to know in real time why Internet connectivity does not work and what they have to do to quickly fix it.

This thesis aims at designing an automated-system to analyse the status of the Internet access in real-time and display information about it to the user through a specific interface. In case of no-connectivity, the system will identify the problem and suggest a possible solution to solve it.

The proposed work is a theoretical design of an automated-system: all processes were implemented in theory without any real discussion about the programming languages that could be used. This work is only a basic documentation to provide information to developers that will have to deal with the feasibility of some processes.

Keywords: Networking technology; internet connectivity; automated system; troubleshooting

Preface

I would like to thank all those who have contributed to my work, especially my advisor Sebastian Sonntag who guided me through structuring the thesis and corrected my work, and my friend Vincent Pesenti who advised me for the contents of my work.

I am also really grateful to my supervisor Professor Jukka Manner who devoted time to steer me in the right direction with my research by providing many reviews and invaluable advice to save me from unnecessary work.

Otaniemi, 30.9.2013

Marine Puibaraud

Contents

Symbols and abbreviations

Abbreviations

ACL	Access Control List
APIPA	Automated Private Internet Protocol Addressing
ARP	Address Resolution Protocol
AUC	Authentication Center
ATM	Asynchronous Transfer Mode
BSC	Base Station Controller
BTS	Base Transceiver Station
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Server
EIR	Equipment Identity Register
ESSID	Service Set Identifier
GERAN	GSM EDGE Radio Access Network
GGSN	Gateway GPRS Support Node
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
HLR	Home Location Register
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
ICT	Information and Communication Technologies
IMSI	International Mobile Subscriber Identity
IP	Internet Protocol
ISP	Internet Service Provider
LAN	Local Access Network
MSC	Mobile Switching Center
NIC	Network Interface Controller
OSI	Open Systems Interconnection
PIN	Personal Identification Number
RNC	Radio Network Controller
RTT	Round Trip Time
SGSN	Serving GPRS Support Node
SNMP	Simple Network Management Protocol
SIM	Subscriber Identity Module
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UMTS	Universal Terrestrial Telecommunications Systems
URL	Uniform Resource Locator
UTRAN	Universal Terrestrial Radio Access Network
VLR	Visitor Location Register
WLAN	Wireless Local Access Networks

1 Introduction

Designed for military purpose in the 70s, the use of the Internet began with interconnecting traditional computers and servers relying on the Internet protocol suite TCP/IP in order to provide services which are currently considered as basic ones, i.e mail, file transfer, browsing data bases, and so forth ([47]). Now, "everything over IP" and "end-to-end" principles remain applied and we can observe that more and more Internet applications are developed to provide new services in daily life, private and professional. This introduction chapter starts with a short description of the current usage of Internet.

1.1 Background

Dependent on the Internet Nowadays, the Internet is more well-known with the World Wide Web, but technologies and devices kept improving in processing speed and increasing traffic capacity [48, Ch1, p28]. Consequently, new trends are appearing such as mobility and turning concrete entities into virtual data. Indeed, as the storage capacity drastically increases, we tend to digitize data contents and store it to get access to it from anywhere with cloud computing ([38]). Concrete objects are disappearing. Information and Communication Technologies (ICTs) tend to offer a wider range of services while Internet applications are becoming more technically advanced than simple mail services, especially multimedia applications.

ICTs must meet the needs of the users who are more and more demanding, as the new expectation is to be connected at any moment, from anywhere. Now clients would like to use their favourite device for business and private life, that is not necessarily a desktop or a cellphone anymore. For instance, a simple house can turn into a whole local access network, that we call "smart house" (See Figure 1) where desktops, the television, the coffee maker, the washing machine, surveillance webcams are interconnected.

1.2 Motivation for the research

Dependence As we explained previously, Internet applications are becoming a vital need. Social networks, electronic-commerce, mobile applications, smart houses are the evidences of such technologies improvement. Besides, with the government's current expectations, getting access to broadband connection is easier than in the last 10 years. Networks are more "intelligent" and provide a better quality of service. For instance, in Finland, as an article from BBC points out [1]:

> Finland has become the first country in the world to make broadband a legal right for every citizen. From 1 July every Finn will have the right to access to a 1Mbps [...] broadband connection. Finland has vowed to connect everyone to a 100Mbps connection by 2015. In the UK the government has promised a minimum connection of at least 2Mbps to

[1] BBC News, 01/07/10, http://www.bbc.co.uk/news/10461048

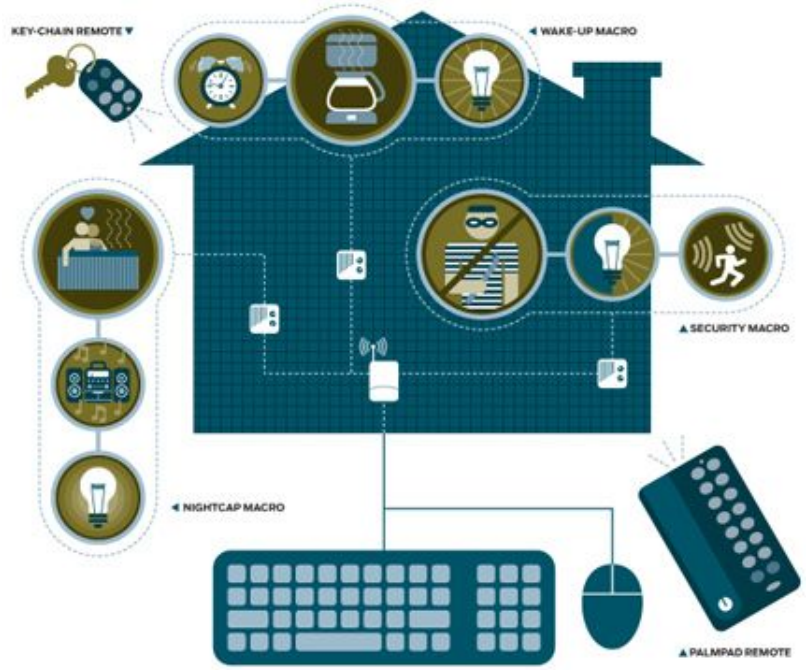


Figure 1: A smart house [1]

all homes by 2012 but has stopped short of enshrining this as a right in law.

Now, any media is turned into a virtual entity so that we can read music, pictures, videos, books on a device, which can be mobile now. Mobility and Cloud technologies are today's challenges while people demand faster and cheaper services [48, p30]. Although home is still the favourite place to get access to Internet, as shown in Figure 2, users demand that access to data and services should be possible any time anywhere. Consequently, as these new technologies are more and more used in professional life, consequences of a short period of no-connectivity to ICTs could be really troublesome for users.

Different users for different usages of Internet We can observe a change in the use of communication devices and also in users classes. Along with the fact that users can get access to a wide range of means of communication, another class of users less expected seems to take its place in the Internet usage: the 60-74 years old. Indeed, as shown in Figure 3, the percentage nearly increased by 4 in seven years. However, all these different classes of users do not have the same knowledges about how to use the Internet services. It is not our intention to evoke a stereotype, but it may be still true that specific classes of users will need more help for the Internet use than others, especially when there are problems with the Internet connectivity. Consequently, users need an *automated* assistant tool to guide them and provide information and instructions in case of something wrong.

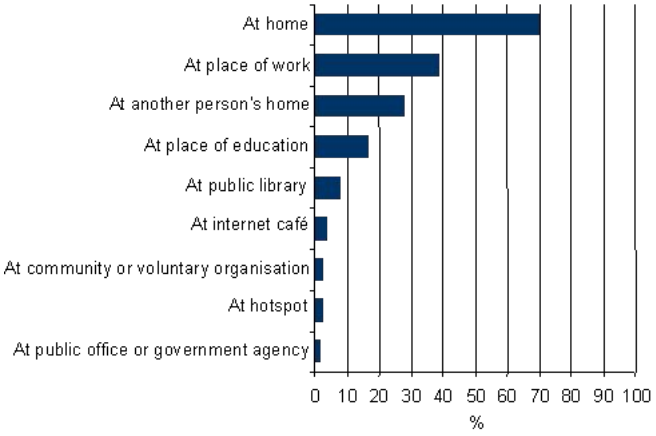


Figure 2: Places of Internet use in spring 2007, per cent of 15-74 years old population [21]

The Internet, a complex network of networks The Internet network involves many operators of different types: protocols, devices, and the management of these components becomes a challenge with the evolution of the Internet technologies and services. A device intended to perform the Internet protocols and applications requires the implementation of the TCP/IP protocol suite, i.e the Internet architecture. All the basic requirements for Internet hosts are explained in [12] and [11], regarding the main different layers and the interfaces between it. As a whole and diverse architecture is involved for the good functioning of the Internet applications, the possible problems that could be the origin of a malfunction of one Internet service could be located in one of the layers of the TCP/IP architecture, regarding a specific component, or protocol.

Network diagnostic tools As we pointed out the fact that the Internet technologies became more like a vital need for users, it can be a real problem if the Internet services do not work anymore. In order to solve such a problem, people designed tools to troubleshoot the problems related to Internet services. Actually, as we explained that the Internet is a wide combination of heterogeneous networks, protocols and technologies, there is also a wide range of problems that could affect the Internet daily usage. The main issue is that all users do not have the same knowledges of the Internet, which is also why we name them "users". In most of the cases, the user works with a tool whose functioning is unknown, and in the event that the tool does not work, the user does not know how to fix it. Consequently, network diagnostic tools have been implemented to deal with such an issue. There are different levels of network diagnosis depending on the type of users it is intended for. Operating systems can already provide such network tools but the problem is that it is not really "user-friendly", and processes may not be automated.

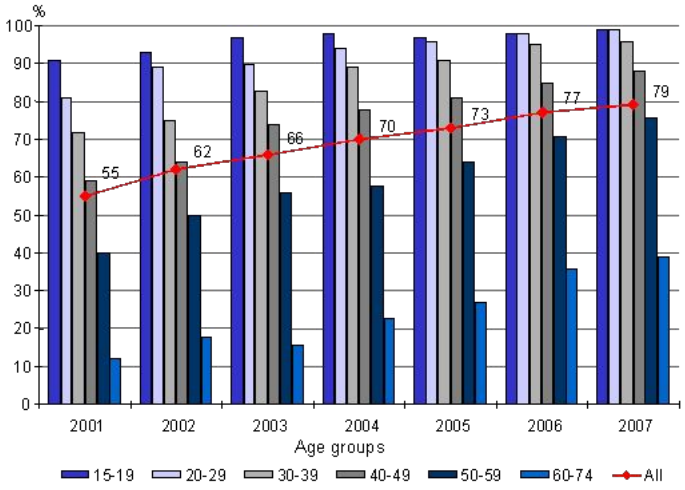


Figure 3: Internet users, spring 2001-2007 [20]

Prevalence of Internet usage and certain purposes of use in 2012

	Used the Internet in the past 3 months	Uses the Internet usually several times a day	Used Internet banking in the past 3 months	Bought over the Internet in the past 3 months	Followed some social network service in the past 3 months	Used the Internet with a laptop outside home and workplace in the past 3 months	Used the Internet with a mobile phone in the past 3 months	Has a smartphone in own use
	Percentage of population aged 16–74 years							
Aged 16-24	100	80	75	54	86	51	51	64
Aged 25-34	100	88	98	67	80	48	48	69
Aged 35-44	98	74	96	61	58	47	47	66
Aged 45-54	96	68	92	54	39	38	38	50
Aged 55-64	82	44	74	31	22	24	24	30
Aged 65-74	61	26	51	13	10	11	11	15
Men	90	66	81	46	44	42	42	54
Women	90	61	83	53	53	32	32	45
Total	90	64	82	49	49	37	37	49

Figure 4: Prevalence of Internet usage in 2012 [23]

1.3 Objective and goals

Because network troubleshooting procedures may not be user-friendly and understandable for users who are not familiar with the Internet technologies, they may need an automated-system that would perform a checking procedure in order to identify the problem, inform the user and may suggest a solution to resolve it. This thesis aims at designing such a system in order to show to the user why the Internet connection is broken. More importantly, this system must be *automated*. In order to implement it, we will proceed the following steps to fulfil the objective. First of all, we will define and classify all the possible main reasons of any kind for which the Internet access does not work. Then we will propose different methods in order to identify and verify what the problem is really about. Finally, we will propose a scheme to implement an automated application that will carry out the automated checking procedure for network diagnosis, then inform the user about the characteristics of the actual trouble, and finally it may provide information to solve the problems, if it is possible.

Purpose of use	All	16-29 yrs	30-49 yrs	50-74 yrs
Sending or receiving e-mails	90	95	91	83
Finding information about goods or services	88	92	92	79
Internet banking	87	84	92	82
Browsing travel and accommodation websites	70	63	75	68
Reading or downloading online magazines	69	68	73	66
Seeking health-related information	62	60	66	57
Obtaining information from public authorities' web sites	56	54	63	49
Looking for information about education, training or course offers	44	57	46	28
Listening to web radios or watching web television	40	53	41	25
Listening to music online or downloading music on PC or other device	39	64	37	16
Reading weblogs	38	53	35	28
Consulting the Internet with the purpose of learning	37	60	34	18
Instant messaging	35	69	28	12
Looking for a job or sending a job application	32	55	31	11
Downloading programmes to the PC	32	45	32	18
Chatting or writing on discussion boards	30	54	26	11
Using browser based news feeds e.g. RSS for reading new content on websites	23	31	23	14
Buying secondhand goods at online auctions or flea markets	23	30	25	13
Internet phone calls	18	19	18	16
Selling goods or services e.g. via auctions	17	22	19	10
Doing an online course	17	29	15	8
Playing games online	14	31	8	4
Subscribing to news services or products to receive them regularly	14	11	17	12
Videoconference	10	12	10	7

Figure 5: Prevalence of Internet usage in 2008 [22]

1.4 Outcomes

In order to fulfil the main objective of this thesis, which is to design a theoretical algorithm in automated mode to target the problems which cause a malfunction in the use of Internet applications or connection, our work led at first to a classification of the different reasons that can turn down the Internet access. After classifying the problems into subcategories that we name *classes of problems* in the next explanations, for each class of problems, we propose a test that can confirm that the analysed class of problem is the actual cause of the Internet malfunction or not. Finally, we develop the automated algorithm for checking the problems in the Internet relying on the tests that would be carried out in a specific determined order. As this algorithm is a global scheme not intended for a particular programming environment, we chose to represent the automated procedure in the form of state diagrams which illustrate the different checking stages to identify the class of problem. This global schematic algorithm should be the basic model for any developers who would like to implement a concrete automated system for network diagnosis.

1.5 Results and Limitations

In this thesis, we tried to consider as many different types of problems as possible, but we admit that depending on the situation, some doubts may remain about

determining which class of problems is the actual origin of a malfunction in the Internet among many suggested problems. Moreover, among the classes of problems we classified, some are external and hidden in the network out of the local access network, which means that our tests cannot reach it.

Admittedly, our work is a theoretical schematic algorithm, so we did not implement it in a programming environment, but we proposed tests based on already existent network tools to target the classes of problems. Actually, we applied some of our tests in some specific situations of problems but all classes of problems we mentioned could not be tested. Indeed, some troubles occur in the external network but we do not have any network administration rights to reproduce such situations to verify that our tests work and confirm the existence of a problem.

1.6 Structure of the thesis

Our thesis structure is organised as following. Chapter 2 focuses on classifying all the possible and most common problems that could enable a malfunction in the Internet usage. In Chapter 3, based on this classification of problems, we propose different theoretical tests to target each problem of our list. Chapter 4 deals with developing the whole automated procedure to perform a network diagnosis, relying on the tests suggested in Chapter 3. Finally, we discuss about the results and limitations we encountered to design such an automated tool in Chapter 5.

2 Classifying the different problems

In Chapter 1 we pointed out the fact that the Internet had become an essential component of the daily life and that due to the wide diversity of users, it was necessary to think about an automated-system that would help them to identify the potential problems in their Internet usage. Consequently, in order to design such a tool, we must firstly review the disturbing problems that are likely to happen while using the Internet services. In this Chapter, we will try to review all these possible problems and classify it into classes of problems. Referring to the explanations about the Internet architecture and components in Chapter 1, we will define the classes of problems. This classification will be necessary to implement the automated checking procedure because it will define a specific optimized order of processing to identify the problem efficiently.

2.1 Background

This section focuses on determining the various reasons that may break the Internet connection or Internet services. Reminding that the network architecture is based on the OSI model [13] and the Internet is formed of heterogeneous subnetworks and protocols, entities are not aware of all the processes functioning in the lower layers. Consequently, from the application layer point of view, it is not obvious to pinpoint the origin of a problem that is happening in the link layer for instance. One of the first difficulty of identifying the problem is to target the layer in which it occurs. Moreover, as Internet is an extended network of networks, a second issue is to find its location in the network.

Admittedly, new technologies and services evolution bring necessarily new technical causes of malfunctioning which may force to update such tool that processes network troubleshooting in order to be able to display every type of problem. However, assuming that among all these various reasons, there are some that are more likely to happen than others: the purpose of this section is then to formulate a classification of the most common network problems so that the automated-system can identify it and may propose a solution to fix it if feasible.

2.1.1 Three main environments in Internet process

In every Internet-based application, whatever the architecture is (Client-Server or Peer-to-Peer ([14])), we can consider three and even four environments interacting in any Internet "process": the end-point source, the end-point destination and the intermediate network binding them, that can be divided in the local access Internet Service Provider network (ISP) and the public Internet, as referred on Figure 6.

Indeed, in applications relying on Client-Server architecture like for instance search engines or simple webservers, a client sends requests to a server through an intermediate network, and the server replies back to the client. Each host can be the source or the destination depending on the type of message, i.e a query or a response. In a Peer-to-peer architecture, the hosts, called *peers*, interact directly

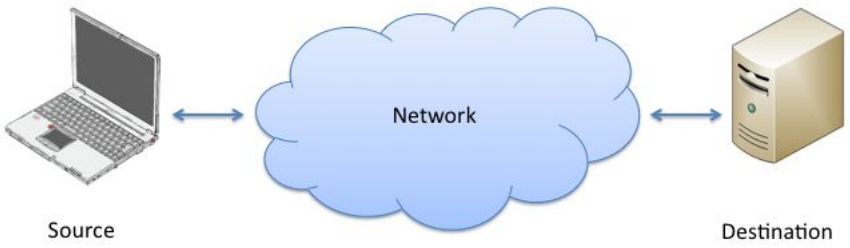


Figure 6: Three entities in Internet process

between each other through a network: each host can be a client and also a "server" for other peers.

2.1.2 Internet connection: step by step

In this paragraph, relying on [29, p531-536], we will explain the different steps to get Internet access when we connect a device to a local access network (LAN) in order to use Internet applications: mail, file transfer, browsing a webpage for instance, considering the Figure 7.

Link-layer connectivity Even though network connection may suggest a virtual connection, it implies to get an access to the network physically. It can be by connecting to a switch for wired LANs, or access points in the case of wireless networks [5], or to the mobile cellular network for a smartphone. Depending on the type of device and the network it is connected to, the interface between the physical layer and the device can be the first reason why the Internet connection does not work.

Get an IP address When the device is physically connected to the LAN, wireless or not, it does not have any IP address, so it needs first to get one. This IP address assignment can be done manually by the user himself or by dynamical mode. Usually, IP address allocation is processed in dynamic mode relying on Dynamic Host Configuration Protocol (DHCP) in most of the cases, but this method is not mandatory [18]. IP address as well as global IP configuration including the IP address of the default gateway and DNS servers (Domain Name System ([34], [35]) is provided from one of the DHCP servers of the LAN the device is connected to. We can note that the DHCP server is not necessarily implemented in the LAN's router. The operating system of the device creates a DHCP request message and broadcasts it on the network. Eventually, the DHCP server replies back a "DHCP ACK message" containing the IP address the device will be identified with, and also the IP addresses of the default Gateway and the DNS server for DNS queries.

DNS In the application layer, the servers and services are identified with a hostname. The structure of the hostname depends on the application, but usually the hostnames are easily understandable by a human being. However, in the IP layer, the devices are identified with an IP address, which is hard to remember for a human user. A simple example is a web browser: it is easier to browse a webpage from a server by typing its Uniform Resource Locator (URL ([9])) address in the web browser bar instead of the IP address. Consequently, a DNS process is required to translate the hostname to an IP address (or also a set of IP addresses) and vice versa.

The DNS process works like as following: an application running on the device needs to reach another computer further in the network, identified with a hostname, whose scheme depends on the application. However, routing the packets is done using the IP address of the destination host, and not its application hostname. As the source does not know the destination IP address, a request is sent to a DNS server to resolve the destination hostname into its corresponding IP addresse(s). The operating system of the device sends a DNS query message to the DNS server. If everything works properly, the DNS server replies back the required IP address or the set of IP addresses that map the destination hostname.

Interacting with the destination computer: Transport and application layer Now that the destination IP address is known, the devices can interact with each other, regarding the transport protocol (TCP ([42]) or User Datagram Protocol ([40])) and application protocols that are used for the service.

NOTE: To simplify the explanation about Internet procedure, we did not take into account security level in this part at the moment. Nevertheless, it is an essential step that could also be a reason why the Internet access is broken, and we will explain it in the next sections with more details.

2.2 The perception of the user face to an Internet problem

Most of usual users of Internet applications do not really know what the Internet is. They tend to be confused with the Internet definition, by not making the distinction between the Internet which is actually the infrastructure and the services its provide, also known as *World Wide Web*. Consequently, they cannot tell the difference between a problem about the Internet connectivity and the one from the Internet-based application itself. For instance, a simple user in business will usually complain about the fact that the network does not work instead of checking that there is in fact something wrong with the application configuration (for instance, mail settings, business software for managing Data bases, etc). For helpdesks in companies, it would be convenient to find a way to filter and classify all these numerous complaints in order to better handle it, as soon as the problem is targeted.

A problem in the Internet access concerns actually a problem in the infrastructure of the network. Referring to the three main environments mentioned in Section 2.1.1, we can assume that there may be a problem in at least:

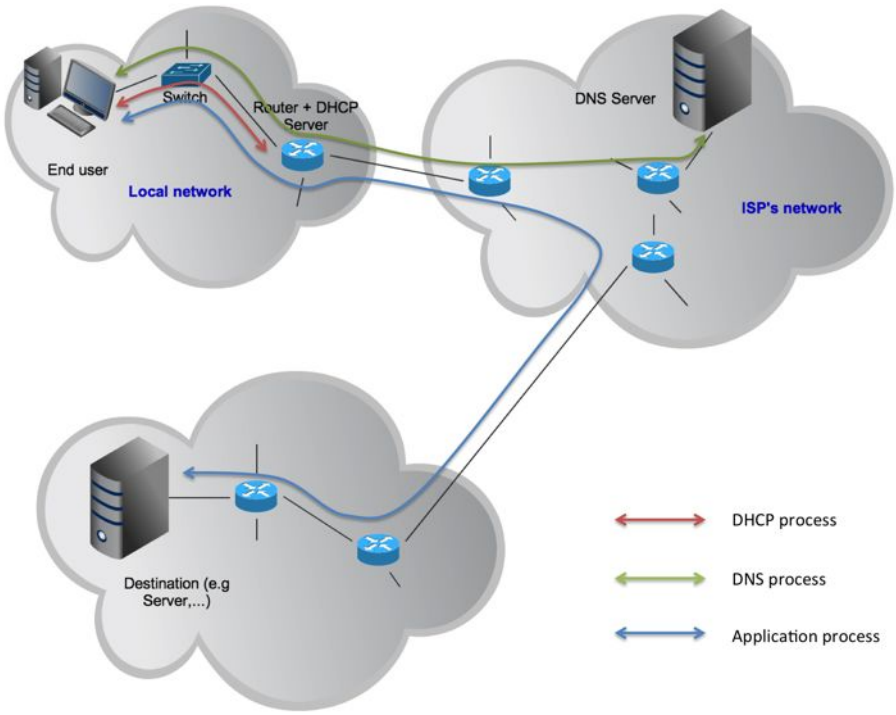


Figure 7: Steps for Internet connection [29, p532]

- The endpoint source, i.e the environment of the user, that we call a *local problem*.
- The intermediate network owned by a third party: formed of the local *access ISP network* and the *public Internet*.
- The *endpoint destination*, also owned by a third party.

In this section, we classify the different possibilities and reasons for which the Internet access does not work, for each of the three entities previously mentioned. Of course, we cannot list all of them, because as a simple user, we may not have knowledges of the environment further than the LAN our device is connected to. We choose this strategy because of the characteristics of each environment. Indeed, the main issue of the user is his lack of knowledges about the functioning of the network further and the destination, more than in the local part. Indeed, these infrastructures are owned by a third-party which also sets protective measures to avoid malicious attacks. Compared to important business companies which own different extended networks, they can measure the Internet performance between their different local networks because of their owner's rights. The simple user can only get some information from one endpoint in his LAN, but he cannot get significant measurement between another endpoint in the public Internet. Moreover, in the LAN, the user has little control over it despite everything.

The scheme of these three environmental components in Internet-based applications highlights the fact that the user cannot impact for network and destination troubles: he cannot fix it himself. This may not be the case regarding some local problems. Obviously, troubles regarding intermediate network and destination cannot be fixed, nevertheless locating the problem remains essential in order to send a report to the responsible institution to fix it. Consequently, locating the problem and providing information about the characteristics of the malfunction is another challenge for the automated system.

Admittedly, the description of our subsystem of three environments is quite brief and these three subgroups are still black boxes whose the different components must be determined. The Figure 8 depicts a more precise representation of what is inside the three subgroups of the system under consideration. In this example, we admittedly displayed a scheme for wired LAN, for instance Ethernet case ([4]), but we would like to design the automated system to function in a standard home network and in mobile cellular networks, that is to say that other network types must be taken into account, such as Wireless Local Access Networks (WLAN, IEEE 802.11), 3G networks ([6]), etc.

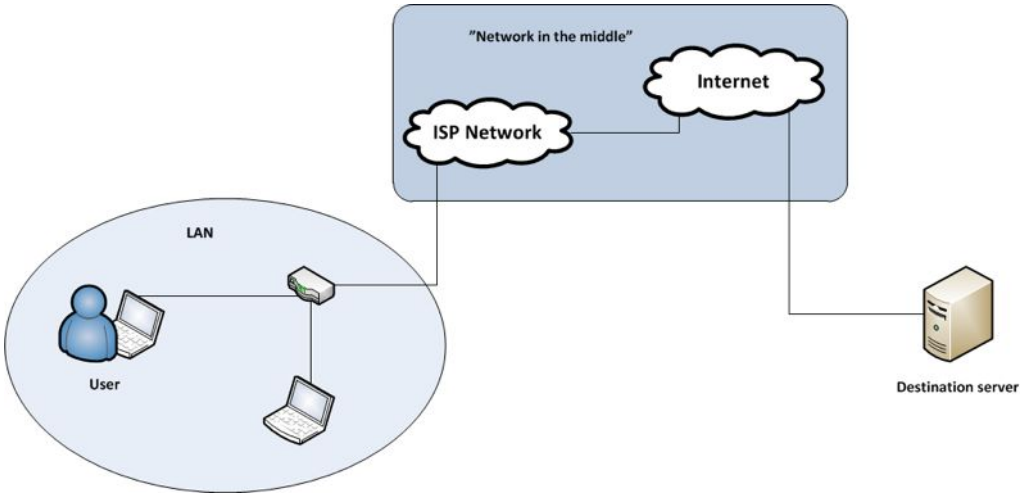


Figure 8: A simplified version of standard Internet access

Next, we will go through the process for determining the different problems and how we classify it.

2.2.1 The user part, or local problems

The user is an unpredictable component in the network, because his behaviour depends on his level of knowledges of the tools that he uses. A good approach should be to check at first if the problem comes from the application itself and not from the Internet connection. For instance, if a user cannot get mails from his e-mail application, it is not inevitably bound to a problem of network access. As we already pointed out this fact, it is one of the usual issues that helpdesks are facing when a user calls to complain: users cannot tell the difference between the application and

the connection. Consequently, before certifying that the Internet access is down, a checking of Internet applications must be done at first. It means that the automated system must confirm that different Internet applications works properly to verify if the problem comes from only one application, and not all. After this process, if indeed, it seems that some Internet applications do not work, it may be possible that the problem comes from the connection.

Generally, the automated system must focus on trying to solve problems in the user part, because a local problem is more likely to happen than some external issue outside. More importantly, this is the only part where some problems can be solved by the user himself. Of course, this is only our assumption based on estimating that local problems are more likely to occur. Then, the automated system must complete an Internet connection checking. A global approach to the issue can be to execute a list of specific processes in a certain order. These procedures must match the different components that take part in the process of Internet connection in every layer that we will explain further:

1. Physical access to the network: Physical Layer and Link layer connectivity: Network Interface Controller (NIC), Switch, Access point, etc.
2. Right of access to the network: Authentication
3. Internet protocols: IP address, DHCP, etc.
4. Internet application layer: DNS process, Internet applications

2.2.2 The environment of the user: the configuration of the LAN

Before focusing on the different processes that must be completed in the user part, the environment of the user must be defined. Indeed, as there is a wide range of possible scenarios, it may be difficult to implement an automated system scalable for heterogeneous situations, that is why we will try at first to tune it only for a certain type of scenarios we describe as following.

The user has few knowledges about ICT technologies, i.e he is not a professional expert. His device is either a computer or a smartphone. The access network he would like to connect to is a standard local network, owned by a standard local ISP. This means that the user did not set himself the configuration of the access network, then DNS server and DHCP server are handled by the operator.

2.2.3 Problem further in the network

If the problem does not come from the user's side, then there may be something wrong further in the intermediate network, or it is also possible that the destination is unreachable. The main issue of this case is that it may be difficult to target the real problem with the few knowledges we have, assuming that ISPs do not provide information about its infrastructure. In this situation, the automated system may inform that there is a problem further, in the destination or the intermediate

networks. However, the challenge of this part is to find a solution to be the most accurate and provide enough information as far as possible.

The following parts will deal in more details with a definition of classes of problems for each subgroup previously mentioned. The Figure 9 depicts a common situation where the user wants to get access to Internet through a local access network, that can be for home, university or business companies for instance. In this scheme, we tried to present every component that could be a cause for Internet connection malfunction.

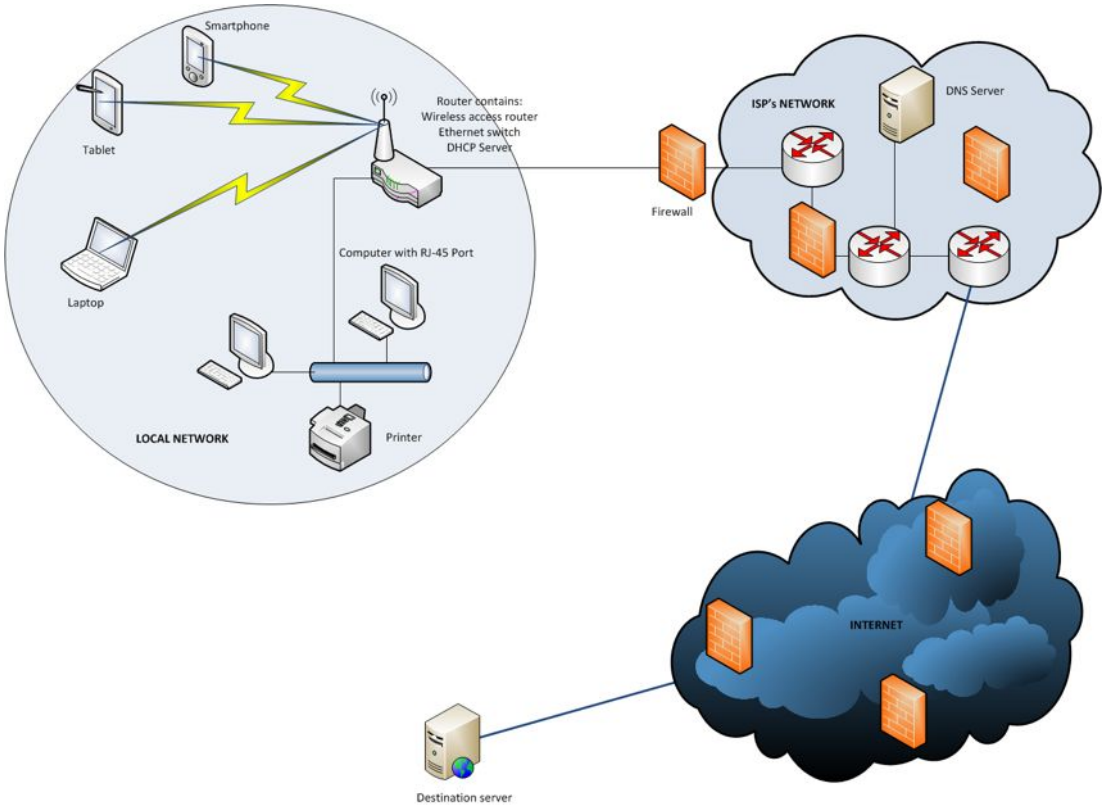


Figure 9: A standard infrastructure of Internet usage

2.3 Wired and wireless LANs: Local problems

In this part, we will expound the spectrum of the different problems of Internet access regarding local problems, that is to say the user and also the local access network. Firstly, we discuss about LAN networks according to the Figure 9, and secondly we will study issues in mobile cellular networks.

There are different methods to classify and order Internet problems. We can organize it according to the distinct involved layers, which is close to the steps of an Internet connection process, explained in Section 2.1.2. The table in Figure 10 displays the classification of the distinct problems for Internet connection regarding the local environment of the user. Each mentioned cause in the table can be commonly related to LAN, WLAN, and cellular networks, except the words displayed

in orange that are exclusively specific to WLANs and green cells meant for mobile cellular networks.

In Figure 10, each column specifies the type of the problem whether it is internal or external. We define *Internal* what is related to the user's device, which suggests that the user can have control or responsibility over it. It is usually about configuring different components in the device to get Internet access. *External* refers to problems that are out of user's control and responsibility, assuming that he is not an administrator of the LAN/WLAN. It implies that in the case of this kind of external trouble, the simple user will not be able to fix the problem by himself.

Local Problems	Internal Problem	External Problem
Physical Layer Link Layer (Physical access)	• No Network Interface Controller (NIC) • NIC problems (settings)	• Incompatible entities • First-Hop Broken
	• No SIM Card	• No coverage
Network Access: authentication	• Wrong credentials	• Authentication server
IP Layer: no IP address	• Wrong settings to get IP address	• DHCP server not reachable • Too many users in the LAN
"Local" Routing		• Gateway problem • Authentication Web • Data Limit
Applications	• DNS solver	
Internet-based Applications	• Wrong settings in software: mail manager	

Figure 10: Classes of local problems

2.3.1 First row: Physical and Link connectivity

This part focuses on the physical and link aspects of the connection to the network. Indeed, a device needs to be "physically" connected to the network in order to send basic traffic i.e bits through it, that is why it implies to have the right equipment to be able to send the data (bits) turned into physical signals to be able to communicate with the other entities in the network. Regarding a computer, the required equipment is the Network Interface Controller (NIC).

Network Interface Controller The NIC is a necessary adapter to connect a device to a network: it implements physical transmission and link-layer services such as framing, link access, flow and error detection [29, p473]. Of course, the adapter depends on the type of network the user wishes to connect to, but nowadays, it is more usual to see that devices are sold with the interfaces already integrated in the device. Basically, a computer has already an Ethernet NIC and a Wireless NIC, that are handled by a specific driver already implemented in the operating system, and the user does not have anything to configure himself.

As it is an essential component of Internet process, a malfunction in the NIC can affect the Internet connection. This malfunction can be due to wrong settings

that would not allow the NIC to communicate with another entity or simply that NIC is broken (electronic flaw, etc.).

The checking process to test if the NIC functions properly may be difficult to implement in the automated-system although testing the NIC is feasible. Indeed, the problem is that operating systems are usually close source, consequently, we cannot get access to the driver functions to use it directly to check the status of NIC.

Incompatible entities Depending on the type of network, the NIC implements different link layer protocols that must be compatible with the first-hop node in order to communicate. Admittedly, NIC of different types are made in the way that the medium is not the same, so that confusion can be avoided, that is why we will not take an interest in this specific case.

First-hop broken This external problem refers to the fact that the connection between the first-hop node and the host is not possible. There are various reasons. In Ethernet case, it can be due to a cable that is broken or not plugged in, an Ethernet switch that does not work properly. In wireless networks, the access can be down and then not visible by the host.

As NIC case previously mentioned, we cannot implement this checking process in the automated-system although it may be feasible. Actually, it may be possible to find that the problem is from this *First-hop broken* class, however, targeting more precisely whether it comes from the medium or the node may not be feasible in our automated-system. For instance, if the problem is in the switch, the automated-system will not be able to confirm whether it is about the medium or the switch, because the switch is hidden from the user, characteristic of the link layer and then acting like a whole link.

No coverage A wireless network that is centralised would present an infrastructure as followings. It can be constituted with several access points among them one will be selected by a mobile station to proceed a connection-establishment because of the best levels of signal and/or bandwidth ([29, Ch6, p564]). Indeed, the mobile station will measure the level of signals of all surrounding access points of the wireless network the user is interested in, and will pick the one with the best signal. Consequently, it may be possible that the connection-establishment fails because of signal levels too low, which means there is no good coverage to get access physically to the wireless network.

NOTE: We make the distinction between the class *No coverage* and *First-Hop broken*, as *First-Hop broken* class is more global and could include *No coverage*. *No coverage* class refers to the first physical link between the end-user and the access point, the first physical node, whereas *First-Hop Broken* is more related to the logical link between the end-user and the gateway in order to transmit IP datagrams over the Internet.

Globally, all these malfunctions in physical and link layer may be targeted in theory, but due to practical limitations especially the non-access to the Operating

System code, functions designed for making the distinction between each of these problems may be difficult to implement. Theoretical methods will be discussed in the next section in order to target each problem in each class.

2.3.2 Second row: Network Access and Authentication class

The user may not be able to send traffic through the network, even though the host have all the required components to connect "physically" to the network and communicate with the first-hop node with the proper protocols. Indeed, the user may not necessarily have the right to send traffic. Most of access networks are not open-access and require an authentication process and other security methods to guarantee that only a certain number of users are allowed to get access to the network.

The main difficulty of this class is to handle the diversity of authentication methods and security protocols implemented. Indeed, in wireless networks, **802.1X** ([30]) is a standard from IEEE that encapsulates Extensible Authentication Protocol (EAP) ([8]): a supplicant host which firstly interacts with an authenticator, which is usually implemented in the Access Point (AP). The host sends a request to authenticate regarding the different security methods advertised by the AP, and then sends to the AP its credentials, that are transmitted to an authentication server, usually using RADIUS protocol [45]. **Captive portal**, also known as web authentication, is another security method to authenticate. We will explain this mode further, as it relies on higher layer protocols.

In this class, internal problems can concern the credentials of the supplicant. The user may not have the correct passwords for instance since old passwords are stored in cache. Moreover, access to some wireless networks rely on authentication protocols that require a specific and additional configuration in the host. It is the case for the eduroam (education roaming) WLAN [2], which is an international roaming service allowing students and researchers to connect to this specific wireless network from various participating sites. If these components are not properly set inside user's device, it may not be able to authenticate to get access to the network. As far as external problems are concerned, something wrong can happen at the third-party authentication, i.e authenticator or authentication server in EAP. This problem may be challenging to target as the application is not aware of what happens in the lowest layers for authentication.

2.3.3 Third row: IP layer components

When the physical and link components of the host are supposed to work, and the host can finally get access to the network and is allowed to send traffic through it, it needs to get an IP address, as we mentioned in section 2.1.2 about Internet connection step by step. There are various methods to obtain an IP address but the most common method is to demand it to a DHCP server through an exchange of requests and replies between the demanding host and the DHCP server.

[2] `http://www.eduroam.fr/`

Wrong settings for IP addressing If the interface of the host is found to be without IP address, there are still various reasons which could explain it. Firstly, regarding internal causes, the user may have set himself the configuration of assigning IP address. Indeed, usually, the operating system leaves the choice between different configuration modes: manually, Bootstrap protocol ([10]), DHCP, etc. Some operating systems such as Windows implements also an automatic IP address assignment when the interface does not have one, called IPv4 Link-Local address (IPv4LL) ([16]), or Automatic Private Internet Protocol Addressing (APIPA). It assigns an IP address in the pool 169.254.0.0/16, but this addressing mode is only useful within a small home network or intranet LAN, not for Internet usage.

DHCP server The reason may come from the DHCP server that is not reachable, because it may be down or the intermediate node between the host and the server is down too. During the process of IP address assignment, messages are exchanged between the host and server. Capturing packets to study the DHCP exchange process may help in theory to tell if the DHCP server is not reachable because of some malfunction or because of the router. We will explain the principle of this test in the Chapter 3.

IP address pool is full because of too many users In certain situations, there may be too many users to handle, so that the DHCP server may reject the host's request for IP address. It will send back a reply "DHCPNAK" to inform the user that its request has been withdrawn [33, DHCP section]. Actually, we can consider this class as a subclass of the *DHCP* class, and indeed for the tests explained in the Chapter 3, we will gather this class with DHCP.

2.3.4 Fourth row: "Local" routing and misleading network access

We named this subsection "Local" routing, but it is not actually appropriate as this denomination refers to the transfer of packets from endpoints to the internal interface of the gateway within the local network.

Router/Gateway problem In the situation in which the host has an interface connected with an IP address but it cannot reach some destination on the outside network, one of the causes can be that the intermediate node between the local network and the Internet does not work, i.e the gateway interface. Probably, helpdesks recommend to check the lights on the set-top box or the router to see if it is still on, and the solution may be to reset the router.

It is difficult to be precise about router bugs, but we can at least classify the problems into two different subclasses:

- The router may be down because of internal bugs that we may not be able to target it because we are not the network administrator. For real, it means we do not receive any reply after pinging the gateway by its IP address.

- The router is up but it cannot route packet outside. It means there is a malfunction with the external interface connected to ISP network. Actually, this particular case is a part regarding routing outside the local network. We consider that it should be more related to ISP problems, and will be discussed in the subsection 2.4 for ISP problems.

Admittedly, it should be convenient to find in which precise component of the router the bug is, but we should not forget that the standard user may only be interested in the fact that there is a problem at the router level, but he does not care about which precise bug.

Authentication Web Authentication Web, also named Captive portal, is an alternative security method mostly used in hotels, airports and restaurants to restrict access to their wireless network [7]. The functioning of this mechanism involves higher layer protocols such as Hypertext Transfer Protocol (HTTP) ([19]). After establishing a connection to the network, the client gets an IP address and IP configuration usually via DHCP. Then the supplicant client is redirected automatically to the authentication captive portal and is invited to enter an identifier and a password on a webpage that has been automatically sent on the web browser. Sometimes, for free wireless networks, the client just has to agree the terms of use on the webpage.

This mechanism can mislead the automated-system, because the host gets an IP address and a proper IP configuration, even before the authentication process is confirmed, which is not the case for EAP. Everything seems to work, but all the traffic is blocked at the authenticator. This class should be actually in *Network Access* category, but due to the misleading IP configuration, it makes assume that the device is able to send IP datagrams and use Internet services. As we classify classes of problems by layers of TCP/IP architecture and also by respecting the order of the steps of an Internet connection, this class problem comes after *getting an IP configuration* class.

2.3.5 Fifth row: Application Layer

DNS solver IP address is the addressing concept in the network layer to communicate, which is not the case for application layer where hostnames are used because it is more understandable with a human point of view. Consequently, translating application hostnames into IP address is an obvious and necessary stage before encapsulating application messages into datagrams and sending it through the network with the corresponding IP address of the destination.

Requests are sent by the user DNS client to ask a DNS server for a hostname translation. The host keeps a list of DNS servers addresses to proceed queries in a specific cache. Usually, the IP address of the DNS server(s) is obtained in the DHCP reply allocating also the IP address to the user. However, it is possible to configure this list oneself using public DNS servers but if these settings are wrong, the DNS client would not get any result for IP address mapping, except an error message telling that the connection timed out and no servers could be reached. This

can mislead about Internet connection status, as the user can interpret this message as a problem in the Internet connection, that is to say, the DNS requests were not transmitted. In fact, network connectivity is up, but the settings for DNS server are wrong, and the requested server was not reachable or was not actually a DNS server.

2.3.6 Sixth row: Internet-based Applications

The last case is not really about Internet connection problem but it highlights the fact that many users do not have enough knowledges about Internet processes and get Internet connection and Internet-based applications mixed up. In this part, we mention Internet-based applications such as mail SMTP, file transfer,...

As Internet applications are really different and work with heterogeneous protocols, the reasons of applications bugs are specific to the application itself. Regarding mail services, it may be because the mail server is not reachable through the network, which is a problem of the category "Problems Further in the network".

Admitting that the causes are numerous and various for Internet applications malfunctions, it is important to highlight that displaying applications problems is not really one of the objectives of the automated-system. However, it is feasible to know if a bug is due to Internet connection or only to a specific Internet application by trying to run different Internet-based applications (web browser, mail software,...) and observe if one of them does not function properly, in which case the bug will only concern the dysfunctional application.

2.4 Wired and wireless LANs: Problems in the local ISP network

In the previous section that deals with *Local problems* category, we tried to list the most usual problems that can happen in the user's environment, depending on the type of local access network, i.e LAN, WLAN and mobile networks. This subsection focuses on problems that occur in another environment out of the user's local environment where he does not have neither knowledge nor responsibility for maintaining the network equipment. In Section 2.1.1, we called it the *intermediate network*, formed with the local access ISP network and the public Internet, as illustrated on Figure 8. The whole Internet is actually a hierarchy of ISPs. At the very top of this hierarchy, we can find a small numbers of tier-1 ISPs interconnected with high speed links. To the contrary, access ISP networks are only at the edge of this network of networks, at the bottom of the ISPs hierarchy [29, p60-61]. This part analyses the problems that are specific to the access ISP network, that we call simply *ISP network*. What we named the *public Internet* is actually the hierarchical network of ISPs that forms the whole Internet. It refers also to the intermediate network between our access ISP network and the targeted destination we want to request. Classes related to the *public Internet* are gathered with the *Internet Destination* environment in the next subsection.

Problems further	ISP Network	Internet Destination
Routing	• ISP Routing	• Internet Routing
Applications	• Firewall	• Firewall
	• DNS	• Destination non reachable
		• Internet performance

Figure 11: Table of classes for "further problems"

2.4.1 First row: Routing in ISP

ISP network routing Firstly, we must assure that the local network works without defect, and that packets can be transferred from an endpoint in the local network to the gateway interface at the border between the local network and the access ISP network. Then this gateway has to route packets further to the next node in ISP network, and so forth, as referred on Figure 9.

What we call *ISP Routing* is a type of class for which the ISP network encounters some trouble to route the packets that the gateway entrusted to. In this class, we take into account two types of subclasses which are:

- The Gateway is not enabled to route the packets outside (we already made a comment for the paragraph *Router problem* in 2.3.4).
- There is a problem further in ISP network. One of the routers of the ISP may not be able to transmit to the next hop.

Dealing with the second point is feasible with difficulty, as the network equipment belongs to the ISP and is protected for security reasons. There may be firewalls to block some ports and protocols to avoid malicious users, that is why trying to get more precise information about ISP network may remain uncertain.

The first point could have been considered with the second as a whole, as it is almost the same case, except that the problem occurs in the first node (router) of the ISP. However, there is another case in WLANs that could mislead the user: the gateway may not be able to route packets outside, because it is not a gateway. Indeed, with the development of wireless technologies, it is easier and more convenient to get access to some devices without plugging-in a cable, such as printers. A Wi-Fi printer is reachable like any other WLAN access points, but when the user connects to the "local network" set up by the printer, the user device gets a proper IP address with a "Gateway" IP address in the routing table. The IP configuration seems correct, except that the printer is not a gateway and cannot perform routing. A test has to be done to check what the characteristic of a gateway is to avoid such mistake.

2.4.2 Application Layer services in ISP

Firewall As security threats get even more complex and widespread over the network, it is recommended to use software-based or hardware-based security such as firewalls to filter the traffic received from or sent to the Internet [28]. Any user can configure a firewall software on his own device to have an additional protection.

It is noticeable that access ISP networks use more firewall and protection in their network to prevent from tracing the routing path of packets from a source host to another destination in the Internet for instance. Indeed, firewalls block some protocols and some TCP ports to avoid flooding and traffic from malicious users [29, p774]. There are three types of firewalls which are traditional packet filters, stateful filters and application gateways. Globally, filtering traffic is based on Access Control Lists (ACLs) [46] that are lists of permissions for hosts depending on parameters such as the IP source or destination address, protocols types, TCP ports and also Internet Control Message Protocol (ICMP) messages [41].

This class of problems *Firewall* could be more classified at the border between the main *Local problems* part and *ISP network* part, as it separates the private local network from the public network. All the traffic coming from the public network and the private network goes through the firewall. This implies that the firewall can block the traffic in both ways, depending on the wishes of the network administrator: for instance, in the case of a standard usage, a user may not get access to a website that would have been blocked by the firewall under the administrator decision. Protocols can be also blocked as well, for instance ICMP ECHO messages that are used by the `ping` ([36]) application to avoid flooding.

Solving hostnames with DNS Usually, the DNS services are provided by the local access ISP, so the user does not need to configure its own DNS server to make a request for a hostname resolution. It can be possible that the DNS services are not available, for reasons that could be external to user device (compared to the internal DNS solver problem mentioned in the section 2.3.5):

- None of the DNS servers of the access ISP network are reachable or down to ISP network matters. The fact that all the DNS servers are unreachable could also be because the intermediate nodes are down.
- The DNS servers of the local ISP network are up and reachable, but none of them can reply with a DNS resolution to a DNS request sent by the client, because the process to resolve the hostname and requesting authoritative servers did not work.

2.5 Wired and wireless LANs: Internet Destination

This part deals with problems that occur further in the *public Internet*, i.e the hierarchical network of ISPs excluding the access ISP of the user, and the *destination* that the user wishes to communicate with. In this case, the connection in the local network access is up and routing in the access ISP networks works too. Nevertheless, the destination may not be reachable. Referring to Figure 8, we may assume that it could be due to the fact that the destination is simply down, or one of the links in the public Internet network between the access ISP network of the user and the destination does not work.

Routing in the public Internet This class of problems is identical to the *ISP Routing* class in the environment of the access ISP network. The different point is that in this case, routing in the access ISP network works properly, but there is the problem in the Internet, i.e the set of nodes pertaining to other ISPs connecting the destination host and the access ISP network of the user. As a matter of fact, we chose to separate this class from the problem of routing in the access ISP network because we want to make a classification of the problems in the perspective to solve it. It means that in order to resolve the problem about network routing, the institutions in charge to be asked to fix the problem are not the same. In the case of access ISP routing, it will concern directly the access ISP network, but in the other case, it is a problem from a tier-ISP higher in the hierarchy.

The main and obvious issue of this class is that depending on the targeted host, the routing path is not the same which means that the fact that a host in the Internet is reachable does not imply that routing in the Internet to another destination works.

Firewalls Like *Routing* class, we already mentioned a Firewalls class in the environment of the access ISP network in 2.4.2. The differences lie in the location of the firewalls as it concerns the ISP networks other than the access ISP network of the user. Actually, the location of the firewalls is a matter of point of view. A host A in a local network protected by a firewall and connected to an access ISP network A wishes to request services from another host B in the Internet. This host B is in fact in an access ISP network B, protected with a firewall too. For the host A, the firewall in ISP network A corresponds to the *Firewalls* class of access ISP, and the other firewall is in the *Firewalls* class related to the *Internet destination*, and vice versa for the host B.

However, more globally, firewalls can be spread all over the Internet, in every ISP network. In some countries, for political reasons, governments impose an Internet censorship: some websites and protocols are blocked so that a user can not get access to. Diverse methods are used such as DNS cache poisoning, standard use of firewalls for traffic filtering, and so on. Consequently, depending on the geographical location of the access ISP network the user gets access to, it may be possible that the protocols or a website he would like to visit are not reachable due to this censorship, and targeting such a problem is really difficult, as it happens even further than in the access ISP network.

Destination not reachable For unknown reasons, although the connectivity between the user device and the destination is up, the destination computer that the user wants to get access to seems to be down. In most of the cases, the user can not fix it as he is not the administrator.

2.6 Internet in cellular networks

This section discusses about problems that can happen in mobile cellular networks.

2.6.1 The architecture of cellular networks

Internet access through mobile cellular networks differ from LANs by the architecture of the local access network. All the classes previously explained except the ones explicitly specific to WLANs are mostly common to LANs and cellular networks, especially regarding the protocols in IP layer and above it. Most of the classes of problems regarding cellular networks that differ from the classes of LANs are part of the *Local problems* main category. Consequently, the next subsection will analyse the problems related to cellular networks, referring to the table of classes for local problems on Figure 10. The table in Figure 11 regarding *Further problems* in ISP network and *Internet destination* is considered as common to cellular networks, since it deals with the public Internet trouble.

Cellular networks have a specific infrastructure to provide Internet access to the subscribers, that is depicted on Figure 12 as following:

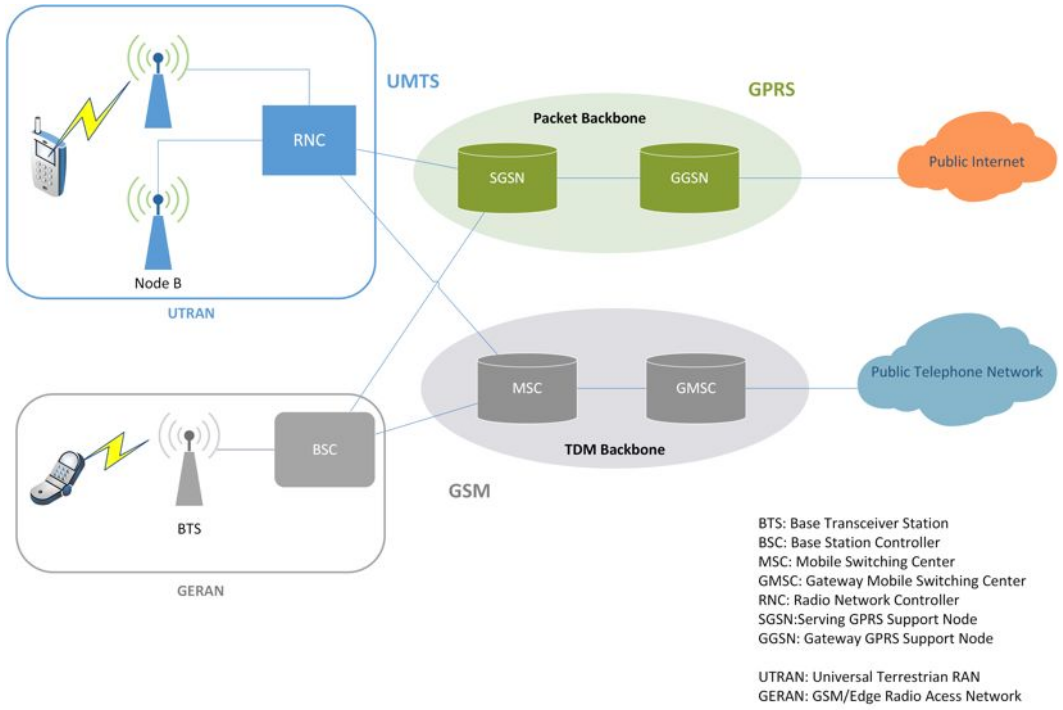


Figure 12: Mobile Cellular Network Architecture (UMTS)

We chose to represent the global Universal Terrestrial Telecommunications Systems (UMTS, ([32])) architecture, also known as 3G cellular networks which provide multimedia services in addition with voice and data services already existent due to Global System for Mobile Communications (GSM [44]) and General Packet Radio Service infrastructures (GPRS, [26]) that UMTS technologies rely on. We did not show EIR (Equipment Identity Register), HLR (Home Location register), VLR (Visitor Location Register) and AUC (Authentication Center) entities on the scheme.

We can observe that IP datagrams can be sent from UTRAN or GERAN interface through respectively Radio Network Controller (RNC) or Base Station Controller

(BSC) to the Switching GPRS Support Node and are transmitted to the Gateway GPRS Support Node (GGSN) to the public Internet. As far as Internet access is concerned in cellular networks, the components that are the most important are the GGSN and the Serving GPRS Support Node (SGSN). Indeed, SGSN authenticates and locates the subscriber, whereas GGSN provides Internet services as IP address assignment, default routing for user equipment, and also authentication based on RADIUS access.

This schema highlights the fact that mobile cellular networks are different from LAN and WLAN architectures, regarding the first three layers of OSI model: physical layer, link layer and network layer. Indeed, specific protocols are used for the transmission between the mobile station and the GGSN: Asynchronous Transfer Mode (ATM, [31]), Frame Relay, Ethernet, etc. We will not analyse in details these protocols but we will globally consider classes of these layers in the *Physical Layer and Link Layer* category regarding our list of classes.

Consequently, except some protocol applications, all the classes in the IP layer and above are common to wired LANs, WLANs and mobile cellular networks.

2.6.2 First row: Physical and Link Layer

Referring to the table of classes on Figure 10, it appears that the problems remain globally of the same kind, even though the technology is different: this is the case for Network Adapter, First-Hop Broken, Network Access (Authentication methods are different since the Subscriber Identity Module (SIM) card is used in the process).

No Network Interface Controller Even though the protocols in the lower layers are different from wired LANs and WLANs, the basic principle of functioning remains the same, as the mobile device needs a network adapter to perform the physical and link layer components. Admittedly, it is exclusively specific to the mobile cellular network technologies, that is why a specific test must be adapted for this NIC.

No SIM Card Actually, the class *SIM Card* could also be classified in *Network Access* category. Indeed, as it stores the International Mobile Subscriber Identity (IMSI), the subscriber identification module is necessary to authenticate the subscribers to get access to its ISP cellular network. Consequently, without a SIM card, the device can not get access to the ISP cellular network. Moreover, the SIM card must be activated with a Personal Identification Number (PIN) code, which is a method to secure the usage of the cellphone. It implies that if the PIN code is wrong, the mobile services are not usable and the cellphone is blocked.

No coverage Like WLAN case, a mobile station will scan all the surrounding base stations of the ISP network it is allowed to connect to, and will choose the one with the best signal. Nevertheless, all areas are not covered by terrestrial networks, and if the mobile station is in one of these isolated and rural areas, it can not get enough signal from a base station to establish a connection.

2.6.3 Third Row: IP Layer

The methods to get an IP configuration and IP address are the same as in LANs, and processed usually with DHCP protocols. Consequently, the problems are equivalent as LANs classes.

2.6.4 Fourth Row: "Local" routing

Gateway problem In mobile cellular networks, the gateway is the Gateway GPRS Support Node. Actually, the GGSN is the main component of the Internet connectivity through cellular networks as it performs many services such as IP address assignment with DHCP protocol for instance, default routing for the user equipment and also authentication process as the GGSN is connected to the Authentication Center, although it is not showed on the Figure 12. Consequently, the classes regarding these mentioned services coincide in GGSN malfunctions, which could be useful to solve any problem of these classes by fixing the GGSN.

Data limit It is difficult to classify this class of problems, as we are not sure of the user settings. Actually, it should be in the *Network Access* category since the user does not have the right to get access to the network, but we considered that it was the same case as *Authentication Web* for WLAN. The configuration is correct, but all traffic going to the gateway is blocked by the ISP policies.

This kind of problem happens when the user reaches for instance the fixed and limited volume of data that he could download in his ISP agreement. Two options are possible for the ISP which can decide to completely cut off the subscriber Internet connection or limit the data throughput. In our work, we consider the first category.

3 Identify and verify the problem in theory

This chapter will focus on proposing a list of theoretical tests meant for computers but it may be intended for mobile phones after adapting it. Based on the classification of the different types of problems from Chapter 2, these tests will be implemented depending on the type of access network: LAN, WLAN, or cellular mobile network. Each test suggested in this chapter aims at:

- Testing a class of problems to verify if the class functions properly or not.
- Then returning a result informing whether the problem comes from the tested class or somewhere else.

A first subsection will deal with analysing tests intended for LAN/WLAN networks, and a second subsection will focus on cellular mobile network diagnosis as it is a very different functioning.

3.1 Common tests to identify the problem

Few basic tests are found on the Web for network troubleshooting, which are commands on the computer terminal. Actually, these commands are really helpful if properly used. The problem as we already mentioned is that the user is interested in an automated tool, and not in a written list of commands that he does not necessarily know how to use or understand. Some of these commands will be used in our tests in the next paragraphs.

Although the topic of the thesis aims at implementing the automated tool for the computer, mobile devices are the next target for a potential and suitable application, that is why it is important to notice that mobile cellphones have different policies of functioning and access to the data of settings can be restricted. Consequently, the system we propose in this thesis is only related to computer devices, although there may be some similarities of functioning with a mobile phone.

3.2 The issue of designing the tests as a simple user

We want to propose an automated system that would make the use of the Internet connection easier for any user, but we are also users of devices equipped with a specific Operating System, that we are not owners of. Indeed, we must take into account that we cannot get access to every entity and data of the device easily, and that some policies restrict our designing. For instance, in the next sections, we will explain that some tests would work in theory, but there are not easily feasible, because the source code of some drivers is not open source.

3.3 Wired LAN and WLAN networks

In this subsection, we will explain how we structured the tests, how we should understand and interpret it, that is to say, which classes are targeted, in which type of network it can be carried out and which device it is meant for.

3.3.1 Structure of the test

Utility of the layered OSI model The principle of functioning of the tests relies on the layered TCP/IP architecture of the network. Indeed, a specific process in a specific layer of the TCP/IP stack can be performed correctly only if everything functions in the lower layers. Considering the list of classes previously enumerated, we will write a series of tests for which each test will target a specific problem in a specific layer. Consequently, depending on the results provided by a test, the automated system will choose another test, and so forth. Consequently, the objective of this Chapter is to define and order the series of tests depending on the results provided by each one.

According to the Figure 13, the results of a test will provide information about the presence or not of a malfunction in the layer. This Figure displays the theoretical TCP/IP stack with the distinct layers that we did not named on the scheme: physical (hardware), network access, internet, transport and application. The test A verifies if the Layer 4 works properly or not. The Layer 4 does not work, so from Test A, it can be deducted that the problem may be in the Layer 4 coloured in red or the lower layers. However, it does not provide any information about higher layers. Consequently, if there are more than one cause in the Internet malfunction, a test must be performed in higher layers. The test B highlights the fact that if a layer (3) properly functions, we can conclude that there is no problem in the lower layers either (2, 1). Finally, the test C checks the layer (5) above the one where the problem is (4). Because the problem is in (4), (5) cannot work properly, which is why the Test C does not work properly either, which makes us suggest from the information displayed in the table that the problem may be in the layer (5) above (4), even though it is not the case.

Observing the results of the test A in Figure 13 would make think that the tests would need to be implemented in a certain order starting with the upper layers and going down in the stack. Indeed, considering the information provided from the Test C on Figure 13, there may be a problem in the layer (5), or in lower layers. Then, the automated system would carry out a test in the layers below (5) such as Test A, that is implemented in the layer (4) that contains the problem, and it will prove that the problem may be in (4), and below. In the same way, the automated system will check the lower layers with the Test B: the layer (3) works properly, which means that there is at least a problem in the layer (4).

In theory, proceeding this way would work if the tests were completely independent from the environment in which they would be carried out. However, in this thesis, the tests are not independent from our field of checking. Indeed, our tests will rely also on the state of the network connection and will be carried out on the device that is meeting network trouble, which means that the results may not be completely reliable. We will propose a solution to get round this problem.

3.3.2 Comparison of the architectures of wired LAN and WLAN

Considering wired LAN and WLAN in the same part We choose to design our tests considering wired LAN and WLAN in a same part, as it presents many

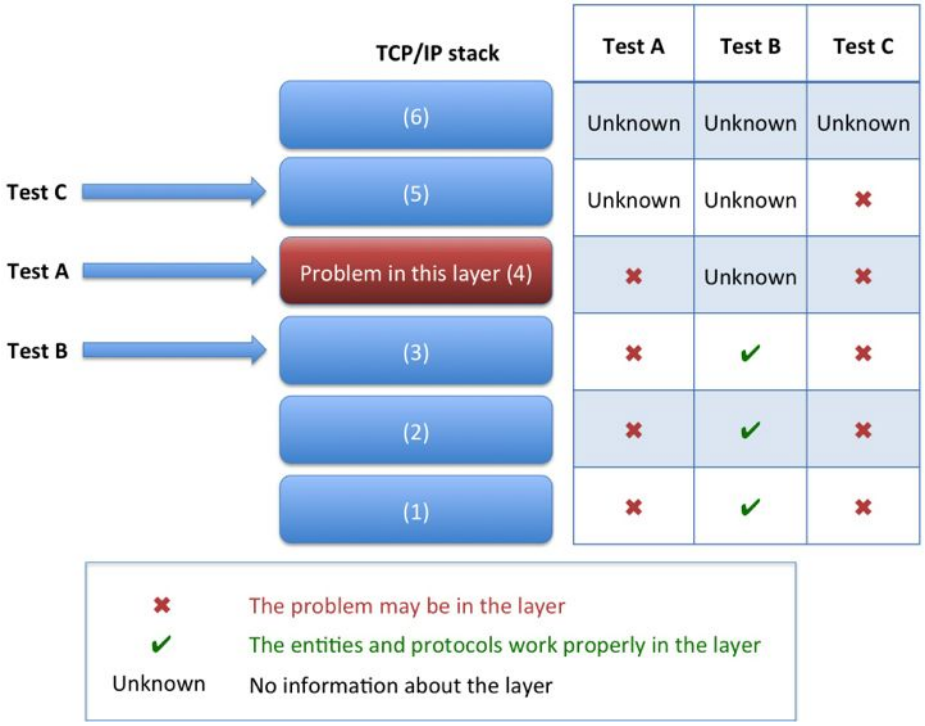


Figure 13: Principle of a test

similarities. We can say that both types of networks differ from the link layer and physical layer in the TCP/IP stack. Indeed, we expect our system to work on a type of standard home network already mentioned on Figure 9, that combined Ethernet and Wi-Fi technologies.

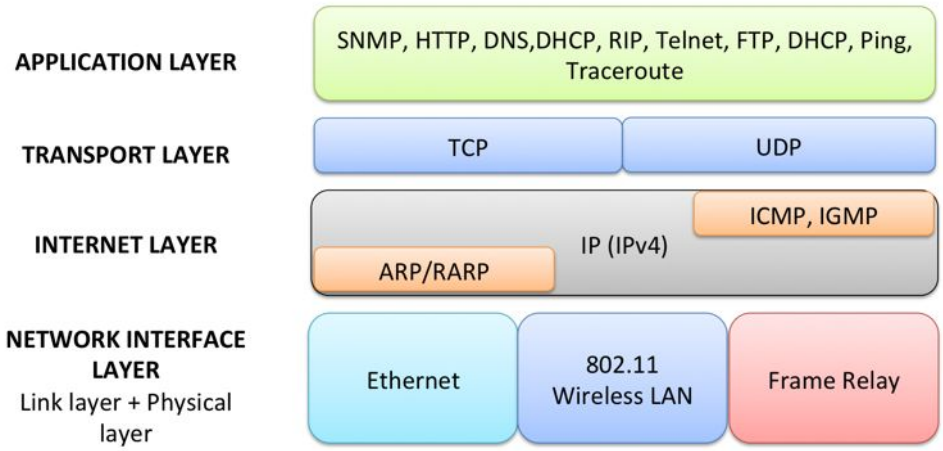


Figure 14: TCP/IP protocol stack

As we can observe on Figure 14, wired LANs based on Ethernet and Wireless LANs differ by the protocols in the network interface layer, that is to say in the link layer (that includes Link Logical Control and Media Access Control), and in the physical layer. Consequently, the automated system will perform common tests for both networks regarding the application layer, transport layer and internet layer, but the problems occurring in the network interface layer will be processed differently depending on the characteristics of the network technology. Actually, those differences affect the implementation of the checking of the NIC class, as it is the entity performing the protocols in the link layer and the physical layer. The test regarding this class will be adapted depending on the type of NIC, intended for either wired LANs or WLANs or even though mobile cellular networks.

3.3.3 Wired LANS: List of tests

This subsection will present the different tests we thought about relying on the table of classes of problems (Figure 10 and Figure 11). This subsection deals with the tests for wired LANs. Due to the structure of encapsulation of processes to transmit an application message from one point to another, we chose to develop our tests globally based on this principle of encapsulation. Our tests are designed in a way that one test will not be enough to identify the problem in one layer with precision. Still based on the principle of encapsulation of processes and the architecture in layered model of TCP/IP, it will be indeed a combination of tests that will delimit the trouble step by step.

How the tests were developed Each test aims at testing a class of possible problems. We already explained that it verifies if the class functions properly or not, and returns a result. The results of each tests are displayed in a table. By interpreting the results, we deduce if the tested class works and depending on its functioning status, we make assumptions about which class is actually the origin of the broken Internet connection. It may be the tested class or another in the classification, which means that other tests must be carried out. For all tests displayed in the tables as following, we interpret the results as if the test has been performed "alone", independently, without executing previous other tests, in order not to be confused with conditioning deductions. Nevertheless, an exception is done with the test *TCP/IP software checking* that will be explained firstly as following.
After listing all our potential tests and analysing the results in the tables, in Chapter 4, we will study a method to combine them in an optimized order to identify the actual cause of a malfunction in Internet applications or connections.

For all classes of problems presented in Chapter 2 (Figure 10 and Figure 11) and displayed in the following tables of tests, we should not forget that each class of problems is a set of "subclasses" of problems. Indeed, for instance, the *DNS class* of problems can concern both the DNS server of the ISP (Third row of the table in Figure 11) and internal problem in DNS solver in the user's device (Fifth row in the table on Figure 10). These two classes are actually subclasses of the main class *DNS*. In order to simplify the presentation of the tests, for each main class, we gathered its subclasses in a new table like Figure 15 and we suggest a suitable global macro test. For each part about a main class, we will study other "micro" tests to target the subclasses when the "parent class" may be the one where the problem is.

The results and deductions from a test will be shown in a table, based on the model of Figure 15. The 3rd column shows the different classes of problem, although there is no mentioning for its subclasses, as it will be discussed in a specific paragraph later. Compared to Figures 10 and 11, we gathered some subclasses and dropped others, as we explained in Sections 2.3, 2.4 and 2.5:

- ➢ *No NIC* and *NIC problems* gathered as *No NIC/NIC Broken*
- ➢ *Incompatible entities* has been dropped, as we consider it as not very common.
- ➢ *DHCP server not reachable* and *Too many users classes* are gathered as *DHCP*.

We ordered the classes of problems mostly basing on TCP/IP stack (2nd column), although the layers are in a reverted order, compared to Figure 13. These classes are gathered in a main group, one of the three main entities in Internet process, mentioned in Chapter 2, presented in Figure 10, and displayed in the 1st column. As we recall it, this subsystem is indeed the basis of the functioning of the automated system to make the Internet connection diagnosis, in a specific order:

- Local Problems (Figure 10)
- ISP Network (Figure 11)

- Internet Destination (Figure 11)

The results of the test are displayed in the last columns. Each column corresponds to a type of result. Depending on a result, we conclude if a class of problem may be concerned. These assumptions are represented as following:

  – If the cell contains a green tick, it means that the corresponding class works properly and the trouble does not come from it.

  – If it contains a red cross, the problem may be in the corresponding class.

The cells are grey when the test does not provide any significant information to decide whether the problem may be in the class or not. Indeed, each test performed in a class only allows us to make assumptions about the class itself and lower classes in the classification, because of the layered model. However, it implies, that another test will be performed to check the classes we do not have any information for. After explaining how we present our tests, the following paragraphs will deal with each of the tests we thought in particular to pick out what the problem is.

TCP/IP software checking Our tests rely on application protocols and tools based on TCP/IP software, then we must at first check that the TCP/IP software is properly implemented in our computer device. Otherwise, the results we will obtain from a `ping` command that we use a lot for our check-list for instance may be wrong because of its incorrect implementation, and not because the requested host is down. Consequently, if we want to avoid any misinterpreting from the results of a test, this checking is necessarily performed at first.
It can be done by pinging the localhost in the device. On most computer systems, the IP address of the localhost is 127.0.0.1 in IPv4. Considering commonly used Operating Systems, as UNIX and Windows, the `ping` command can be used to perform this test. However, if this command does not exist on the device system, such equivalent command could be implemented. Actually, it is noted in [12, p42] that

> *Every host MUST implement an ICMP Echo server function that receives Echo Requests and sends corresponding Echo Replies.*

`Ping` is a network tool based on ICMP protocol, with Echo request and Echo reply messages. We choose to implement this test by using the command in a terminal:

```
ping 127.0.0.1
```

The Figure 15 displays the information provided by the results of the test. Usually, if we can read that there are reply packets from the localhost (what we named "Correct reply" in the result), the TCP/IP software is properly implemented: we can use applications and protocols based on TCP/IP architecture, as well as some network commands for our troubleshooting without misinterpreting, for instance

pinging other hosts. However, if there is a problem in the TCP/IP software, we will get an Error message or no reply from our ping command, as showed in the column "No reply" in the table. With this test, there is no doubt about determining whether the actual problem comes from TCP/IP software or not. TCP/IP implementation is independent from the Internet connection, so if it does not work, it is only related to TCP/IP implementation. However, we should not exclude the fact there might be more than one problem, higher in the layer architecture, but this case can not be checked with this test only.

		Test implementation	Test TCP-IP software Ping localhost (127.0.0.1)	
		Result of the test	Correct reply from localhost	No reply
		Type of problem		
LOCAL PROBLEMS	TCP/IP Stack implementation	Wrong installation: reset TCP-IP stack	✔	✗
	PHYSICAL LAYER, LINK LAYER	No NIC/NIC broken		
		First-hop broken		
	NETWORK ACESS: authentication	Wrong credentials		
		Authentication server		
	IP LAYER: No IP address	Wrong settings to get IP address, IP configuration		
		DHCP		
	LOCAL "Routing"	Gateway problem		
		Authentication Web		
ISP NETWORK	ROUTING	Routing		
	APPLICATIONS	Firewall		
		DNS		
INTERNET and DESTINATION	ROUTING	Internet routing		
	APPLICATIONS	Firewall		
		Destination no reachable		
		Internet Performance		

Figure 15: Test for the TCP/IP software

IP configuration checking This checking should be performed just after the TCP/IP software checking to find out if the computer device has proper IP settings to send data over the Internet, that is to say:

- A correct IP address
- The IP address of the default gateway to route packets over the public network
- The IP address of a DNS server to solve hostnames

If one of these entities is missing, the user cannot use Internet-based applications. In most of common cases, the IP configuration can be done in dynamic mode or

manual mode. We characterize an IP address as correct when there is no IP address conflict with another host in the local network, and that was not obtained from the particular automated private addressing process APIPA mentioned in Chapter 2, and that it can be used for Internet protocols.

Then the global test *IP configuration checking* consists in firstly verifying which method is used to set an IP configuration. If the IP addresses have been set manually, the next step will be to check that the IP address is not used by another device in the local network, and that the addresses of the gateway and DNS server are correct, otherwise, the automated system will inform the user. The interesting part is when the way to get the IP configuration is set by dynamical mode DHCP (we will consider only this method for dynamical mode).

After confirming the mode of getting an address is dynamical, we can display the IP configuration, still relying on Windows and UNIX OS, using the command `Ipconfig`[3]/`Ifconfig`[4] depending on which OS is used. The routing tables can also be verified to make sure that the default gateway is properly set, otherwise there may be something wrong in TCP/IP settings, or that there is no connection. The purpose of this test is to make sure that the computer device has all the required elements previously mentioned to carry out Internet-based applications.

On Figure 16, we presented the results as if this test has been performed alone, after the TCP/IP software checking. As we can notice, one test is not enough to isolate the problem. Indeed, the fact that there is no IP address (2nd column of the results) makes us assume there may be an erroneous function in one or more of the classes in local problems, which is not precise. However, performing this test secondly could make the procedure more efficient, because it will lead us to directly perform a next test in:

- Either ISP network, if the IP configuration is correct
- Or Local network if there is no IP address

The 3rd column of the results is a particular case where the interface has an IP address but the IP address of the default gateway in the routing table is wrong or does not allow the device to send IP packets. We consider this case as less regular, but we will take it into account in the part regarding ISP routing problem.

Network Interface Controller Checking The functioning of Network Interface Controller that implements physical layer and link layer is usually a closed-source software provided in most of operating systems. However, due to the fact that the NIC is controlled by the NIC driver, its status can be analysed through a user interface to check if it encounters a problem. Depending on NIC models, it is also possible to check that the NIC is working by looking at the LED lights, if it is a solid green light or flashing, informing the user that the NIC is connected and is receiving or sending data. Nevertheless, these are external indicators for a user.

[3] `http://net-tools.sourceforge.net/man/ifconfig.8.html`
[4] `http://technet.microsoft.com/en-us/library/dd197434(WS.10).aspx`

		Test implementation	IP address settings checking (DHCP, manually, gateway) Ifconfig/Ipconfig		
		Result of the test	IP address OK IP address of Gateway OK	No IP address	Routing table: default route wrong
		Type of problem			
LOCAL PROBLEMS	TCP/IP Stack implementation	Wrong installation: reset TCP-IP stack			
	PHYSICAL LAYER, LINK LAYER	No NIC/NIC broken	✔	✗	
		First-hop broken	✔	✗	
	NETWORK ACESS: authentication	Wrong credentials	✔	✗	
		Authentication server	✔	✗	
	IP LAYER: No IP address	Wrong settings to get IP address, IP configuration	✔	✗	✗
		DHCP	✔	✗	✗
	LOCAL "Routing"	Gateway problem		✗	✗
		Authentication Web			
ISP NETWORK					
INTERNET DESTINATION					

Figure 16: IP configuration checking

Although we do not have a concrete scheme to implement such a test, the automated system would have to execute a process to check the status of NIC to find out if there is a flaw in the NIC, both in hardware and software. This process can work, based on NIC driver implementation and can fetch information such as the status we can observe on the NIC interface with the NIC driver.

		Test implementation	NIC Diagnosis NIC Driver to analyse NIC status	
		Result of the test	NIC works properly	There is a problem in NIC
		Type of problem		
LOCAL PROBLEMS	TCP/IP Stack implementation	Wrong installation: reset TCP-IP stack		
	PHYSICAL LAYER, LINK LAYER	No NIC/NIC broken	✔	✗
		First-hop broken		
	NETWORK ACESS: authentication	Wrong credentials		
		Authentication server		
	IP LAYER: No IP address	Wrong settings to get IP address, IP configuration		
		DHCP		
	LOCAL "Routing"	Gateway problem		
		Authentication Web		
ISP NETWORK				
INTERNET DESTINATION				

Figure 17: Test for the NIC

On Figure 17, we simplified the table by not showing all the classes, as only the classes in local problems are concerned. This theoretical test would reveal if the NIC works properly, consequently, the problem does not come from the NIC, or it would highlight a bug in NIC hardware or software. The test would be really specific to

troubleshoot the NIC, and does not provide any other information for another class. As we recall it, the difficulty of such a test is to design it as the implementing code for NIC may be closed-source.

First Hop Checking The First-hop broken class may concern many subclasses of problems, such as a cable that is not properly plugged-in or that is broken, or a switch/hub that is down in wired LANs. Cable testing for instance is a delicate issue as it usually needs external material, like a tone generator and amplifier pair. All these different reasons are difficult to highlight one at the time, but as they all belong to the physical layer and link layer, under the network layer, we can use Address Resolution Protocol (ARP [39]) to check that the links are on. ARP protocol is used to translate IP addresses into MAC addresses in the local network, and these translations are stored in an ARP cache in the computer device. We can display the ARP cache to verify if there is at least one entry matching the MAC address and IP address of the gateway. Each entry has a Time To Live value, but it is possible to update any entry. If the arp cache is empty, it means there was no ARP request sent from our computer to ask for an IP address translation, which would be unusual, as the gateway is at least requested for routing: an empty ARP table may suggest that there is no connectivity because the link may be down. Nevertheless, it is more difficult to find out for which reason the link is down amongst the subclasses we discussed about at the beginning of the paragraph.

The Figure 18 shows where the problem could be if we perform this test independently.

		Test implementation	First-Hop Checking Verify that there are entries in ARP table (at least Gateway)	
		Result of the test	At least the Gateway entry	No entry
		Type of problem		
LOCAL PROBLEMS	TCP/IP Stack implementation	Wrong installation: reset TCP-IP stack		
	PHYSICAL LAYER, LINK LAYER	No NIC/NIC broken	✔	✗
		First-hop broken	✔	✗
	NETWORK ACESS: authentication	Wrong credentials	✔	✗
		Authentication server	✔	✗
	IP LAYER: No IP address	Wrong settings to get IP address, IP configuration		
		DHCP		
	LOCAL "Routing"	Gateway problem		
		Authentication Web		
ISP NETWORK				
INTERNET DESTINATION				

Figure 18: Test for First-Hop related problems

ARP protocol can still be used to check that the links are up/down by listening in passive mode to ARP broadcast requests from the other local users. Such a tool is proposed like `Netdiscover` ([37]), a network scanner based on ARP protocol.

Two modes can be used: active mode enables ARP requests from the user, whereas passive mode consists in listening to broadcast ARP requests from other users. The purpose is the same as capturing ARP packets on an interface with a packet analyser, but the tool is simple of use.

```
netdiscover -r {IP address pool} [-p]
```

The option `-p` is to enable passive mode. This instruction carries out an automated scanning of the hosts that are up in the local network for this pool of address. Admittedly, when we do not have information about the network, we can try different pools of IP addresses as we know that usually private IP addresses are used.

Finally, a third method could consist in listening in passive mode any packet from all protocols that the interface receives. We had a short experiment to confirm the feasibility of this test. We connected our laptop to a wired Ethernet network of our student house that does not need authentication. In this network, the IP allocation is DHCP-based, but we manually set an IP address in the private addressing, for instance 10.0.0.3. Then we captured all packets on the corresponding interface while we were trying to ping the gateway, without success obviously. From the packets, we observed that the interface did receive packets, and that the device received an ARP reply from the Gateway after asking for translating the IP address of the Gateway into MAC address. Among packets capture, we noted that most of packets received were broadcast packets such as Spanning Tree Protocol [5]packets, meant for switches or also Logical Link-Control protocol [6]. The fact that the interface receives packets proves that the device is connected according to Physical Layer and Link Layer, but it is not allowed to use higher protocols. Indeed, we tried to ping the gateway with an incorrect IP source (10.0.0.3), and it did not work. In a global situation, if we do not receive any packets on the interface, it means that there is a problem in *First-Hop* Class or *Network access* is not allowed.

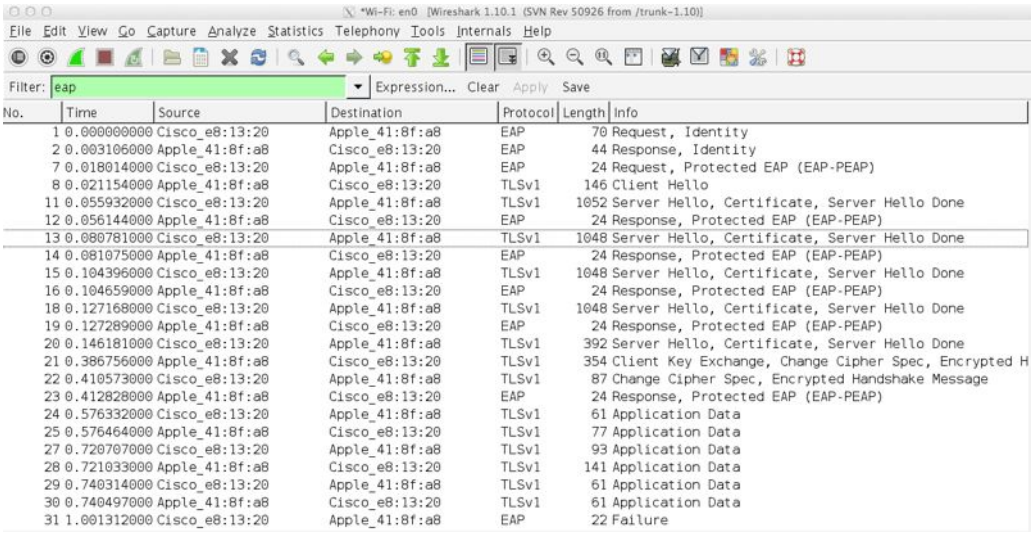
Authentication Checking Authentication Checking is a delicate issue because there is no explicit process to verify it. Usually, when a user wants to connect to a network, he is invited to provide its credential to get access, and in case of failure, a message is displayed on the screen to inform that the authentication process succeeded or failed. In this work, we assume that authentication procedure is based on EAP-TLS security protocol. During the authentication process, the supplicant (the user) and the authentication server exchange messages, among which there is a key exchange in order to establish a secure communication later. In the event of a failure, the supplicant receives a failure EAP packet from the authentication server.

A way to know that the authentication is required and then may have failed or succeeded is to capture packets from EAP-TLS protocol to analyse the exchange procedure, using a packet analyser. In this way, this test is less uncertain in the

[5]`http://www.cisco.com/en/US/tech/tk389/tk621/technologies_configuration_example09186a008009467c.shtml`

[6]`http://www.javvin.com/protocolLLC.html`

extend that we can determine if the problem is due to authentication by the type of packet we analyse. The Figure 19 displays a packets capture on wireshark [7] during an attempt to connect to a network, in this case a WLAN. After asking the identity of the supplicant Apple, the authentication server Cisco requires the TLS client certificates that the supplicant tries to provide without success (Client Key Exchange packet), which ends in failure. The client is withdrawn from the network, and cannot receive network traffic anymore, even broadcast.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	Cisco_e8:13:20	Apple_41:8f:a8	EAP	70	Request, Identity
2	0.003106000	Apple_41:8f:a8	Cisco_e8:13:20	EAP	44	Response, Identity
7	0.018014000	Cisco_e8:13:20	Apple_41:8f:a8	EAP	24	Request, Protected EAP (EAP-PEAP)
8	0.021154000	Apple_41:8f:a8	Cisco_e8:13:20	TLSv1	146	Client Hello
11	0.055932000	Cisco_e8:13:20	Apple_41:8f:a8	TLSv1	1052	Server Hello, Certificate, Server Hello Done
12	0.056144000	Apple_41:8f:a8	Cisco_e8:13:20	EAP	24	Response, Protected EAP (EAP-PEAP)
13	0.080781000	Cisco_e8:13:20	Apple_41:8f:a8	TLSv1	1048	Server Hello, Certificate, Server Hello Done
14	0.081075000	Apple_41:8f:a8	Cisco_e8:13:20	EAP	24	Response, Protected EAP (EAP-PEAP)
15	0.104396000	Cisco_e8:13:20	Apple_41:8f:a8	TLSv1	1048	Server Hello, Certificate, Server Hello Done
16	0.104659000	Apple_41:8f:a8	Cisco_e8:13:20	EAP	24	Response, Protected EAP (EAP-PEAP)
18	0.127168000	Cisco_e8:13:20	Apple_41:8f:a8	TLSv1	1048	Server Hello, Certificate, Server Hello Done
19	0.127289000	Apple_41:8f:a8	Cisco_e8:13:20	EAP	24	Response, Protected EAP (EAP-PEAP)
20	0.146181000	Cisco_e8:13:20	Apple_41:8f:a8	TLSv1	392	Server Hello, Certificate, Server Hello Done
21	0.386756000	Apple_41:8f:a8	Cisco_e8:13:20	TLSv1	354	Client Key Exchange, Change Cipher Spec, Encrypted H
22	0.410573000	Cisco_e8:13:20	Apple_41:8f:a8	TLSv1	87	Change Cipher Spec, Encrypted Handshake Message
23	0.412828000	Apple_41:8f:a8	Cisco_e8:13:20	EAP	24	Response, Protected EAP (EAP-PEAP)
24	0.576332000	Cisco_e8:13:20	Apple_41:8f:a8	TLSv1	61	Application Data
25	0.576464000	Apple_41:8f:a8	Cisco_e8:13:20	TLSv1	77	Application Data
27	0.720707000	Cisco_e8:13:20	Apple_41:8f:a8	TLSv1	93	Application Data
28	0.721033000	Apple_41:8f:a8	Cisco_e8:13:20	TLSv1	141	Application Data
29	0.740314000	Cisco_e8:13:20	Apple_41:8f:a8	TLSv1	61	Application Data
30	0.740497000	Apple_41:8f:a8	Cisco_e8:13:20	TLSv1	61	Application Data
31	1.001312000	Cisco_e8:13:20	Apple_41:8f:a8	EAP	22	Failure

Figure 19: A packet capture during a connection attempt

Although it may be an efficient method, it is however a controversial option, as a packet analyser is restricted-use for security and private reasons. In this case, the usage of such a tool should be limited only to listening to packets intended for the user only. As a matter of fact, the network interface does receive and sends all the traffic that we can observe on a packet analyser, consequently a developer who would like to implement such test could design a program that would be an equivalent of a packet analyser but with more filters to restrict the packets to the specific ones needed forr this test.

However, if it is not possible to use a packet analyser, another option to test if the problem comes from authentication is to proceed by process of elimination of classes of problems: if everything works in the physical layer (links, NIC) and the device does not have an IP address, then it may be because of an authentication problem.

DHCP Checking The same approach as *Authentication checking* can be performed, using packet capture to observe if an exchange of DHCP messages occurs between the user device and the DHCP server. This can be done listening to packets on UDP ports 67 (DHCP requests from the client) and 68 (DHCP replies from the server)[29, p381-385].

[7] `http://www.wireshark.org/`

		Test implementation	Authentication Test Tracing packets of authentication protocol with a network protocol analyzer	
		Result of the test	Authentication success messages	Authentication failed
		Type of problem		
LOCAL PROBLEMS	TCP/IP Stack implementation	Wrong installation: reset TCP-IP stack		
	PHYSICAL LAYER, LINK LAYER	No NIC/NIC broken	✔	✔
		First-hop broken	✔	✔
	NETWORK ACESS: authentication	Wrong credentials	✔	✗
		Authentication server	✔	✗
	IP LAYER: No IP address	Wrong settings to get IP address, IP configuration		
		DHCP		
	LOCAL "Routing"	Gateway problem		
		Authentication Web		
ISP NETWORK				
INTERNET DESTINATION				

Figure 20: Test for authentication related problems

As DHCP procedure was explained in Section 2.1.2, to confirm that the device got an IP address, it must receive a DHCP ACK from a DHCP server: it is the column "DHCP request was successful" in Figure 21. Regarding the result "No DHCP reply", the reasons why DHCP replies are lacking in the packet capture could be that none of the DHCP servers are reachable, because of a defect in the DHCP servers or the intermediary gateway itself which does not transfer packets to the servers. Even if it is more complexed to determine which cause is true, the main problem for the user is that he cannot get an IP address and he cannot fix it if he is not a network administrator.

The result "DHCP request failed" corresponds to the case when the client could not get an IP address from the DHCP server. This situation is observed when after making a request for an IP address lease to the DHCP server, the client receives a DHCP NAK telling that the server has withdrawn its request. This may happen if there is an over-allocation addresses due to too many users connected [33]. Actually, DHCP NAK message can correspond to different situations regarding the IP lease the client is requesting: the client would like to ask for a new lease, or get a reallocation for its current one after rebooting, or renew it or eventually rebind it. In all the cases the client receives such DHCP NAK, he comes back to an initial state to ask for a new lease, because the server refused to provide the right to use an IP address.

Gateway Checking This test aims at verifying if the gateway is active or down by pinging its IP address. As well as for *TCP/IP software checking*, if the ping command does not exist on the device, an equivalent tool can be implemented based on ICMP protocol to send a request to the gateway and wait for its reply. In the event that there is no reply, the problem may be in the gateway or in lower classes,

		Test implementation	DHCP Server Test Tracing packets of DHCP protocol with a network protocol analyzer		
		Result of the test	DHCP request successful: IP address provided	DHCP request failed	no DHCP reply
		Type of problem			
LOCAL PROBLEMS	TCP/IP Stack implementation	Wrong installation: reset TCP-IP stack			
	PHYSICAL LAYER, LINK LAYER	No NIC/NIC broken	✔	✔	✗
		First-hop broken	✔	✔	✗
	NETWORK ACESS: authentication	Wrong credentials	✔	✔	✗
		Authentication server	✔	✔	✗
	IP LAYER: No IP address	Wrong settings to get IP address, IP configuration	✔	✔	✗
		DHCP	✔	✗	✗
	LOCAL "Routing"	Gateway problem			
		Authentication Web			
ISP NETWORK					
INTERNET DESTINATION					

Figure 21: Test for DHCP related problems

as showed on Figure 22.

This test can be properly interpreted only if the IP address of the gateway is known in advance and correct. If the user is familiar with the network in which the gateway has a static IP address, the test can be done pinging the IP address of the gateway. Nevertheless, in the event that he would like to connect his device to a network that he does not know, and where the IP address allocation is down by DHCP, he can not know the IP address of the gateway without obtaining an IP address from the DHCP server.

Consequently, we admit that the results displayed on Figure 22 are related to a more particular situation, although we wanted to respect the same scheme of presentation for each test in this section, as if the test has been carried out alone. We assumed in this case that we knew the IP address of the gateway in advance, without DHCP server, which signifies that the main class "*No IP address*" and all the lower classes can be still considered in the event of an absence of reply from the gateway. However, as we will explain the main procedure of tests of the automated system in more details in the next section, this test will be performed after confirming that there is no problem in the lower classes, which means the device was able to get a proper IP address and configuration then.

Let us make a short comment about the fact that the class "Authentication Web" is much more related to WLANs, however we added it in LANs, in order to get a global view of the automated tool for both types of networks. Nevertheless, in the tests exclusively related to LANs, the automated system will not take into account this class.

ISP Routing Checking Considering that the local network works properly, and the packets from a user can reach the gateway, it is afterwards the responsibility

		Test implementation / Result of the test / Type of problem	Gateway checking: Ping Gateway — Reply from Gateway	Gateway checking: Ping Gateway — Request timed out
LOCAL PROBLEMS	TCP/IP Stack implementation	Wrong installation: reset TCP-IP stack		
	PHYSICAL LAYER, LINK LAYER	No NIC/NIC broken	✔	✗
		First-hop broken	✔	✗
	NETWORK ACESS: authentication	Wrong credentials	✔	✗
		Authentication server	✔	✗
	IP LAYER: No IP address	Wrong settings to get IP address, IP configuration	✔	✗
		DHCP	✔	✗
	LOCAL "Routing"	Gateway problem	✔	✗
		Authentication Web	✔	✗
ISP NETWORK				
INTERNET DESTINATION				

Figure 22: Test for Gateway related problems

of the ISP to route the packets from its own network to another network until it gets the desired destination. This test consists in highlighting a possible problem in ISP network which would explain why the user can not get access to a specific destination, because of a routing problem in ISP network. If the problem does exist within the ISP network, it is not possible to know precisely what is the cause and obviously nothing can be bone to fix it.

We want to target the routing problem in ISP network by pinging different destinations in the Internet, whose IP addresses remain the same, and then are easily to remember, such as public DNS servers. For instance, Google advertises two IP addresses for its DNS servers: 8.8.8.8 and 8.8.4.4 (IPv4)[8]. Other public DNS servers are reported on some websites like [2]. In this case, the automated system should update the IP addresses of the public DNS servers to ping for this test.

The Figure 25 sums up the results obtained if the test is performed independently, except the TCP/IP software that has been checked firstly. For instance, the automated system can ping both public DNS servers of Google. If we get a reply from at least one of the requested servers, the problem is not related to ISP routing.

Nevertheless, the fact that there is no reply from any of the servers leads to three possibilities of trouble:

1. It could be explained by the fact that the servers are down. A proper method should be to ping different public DNS servers from different owners, for instance: 8.8.8.8 from Google, 212.30.96.211 located in France and 213.115.183.13 in Sweden, Kista (updated one month ago and fetched from the website [2]).

2. Admittedly, it can be possible that all servers from different locations and owners are down, even if the probability is really low, but we can still fix

[8] `https://developers.google.com/speed/public-dns/?hl=fr`

a number of pinging requests that the tool will carry out until it takes the decision to confirm that the problem is due to a wrong functioning in ISP routing, and not because the servers are down. The cause could be because some node within the ISP network or further in the public Internet is not operational.

3. The other case still regards ISP routing, but at the first node, which is the gateway between the local network and ISP network. The gateway could have a similar malfunction as in the second point, but it could be possible that it is not a gateway that has an equivalent functioning as a WiFi access point, which misleads the user device. Admittedly, this case only occurs in WLANs, and will be discussed in the part related to WLANs tests.

It is important to notice also that every result from a ping command should be analysed carefully regarding the values of the Round Trip Time (RTT) and the packet loss rates: a significant RTT and a packet loss rate different from 0% could mean that the Internet performance is affected, although Internet connectivity works. Nevertheless, Internet performance is the last class of problem to be discussed that is why, we only reported two types of replies from ping command which are: "there is a reply" and "there is no reply at all".

As well as the other tests, it does not make sense to perform this test alone, since it only adds even more classes of problems to take into account as we climb up the classes stack ordered like TCP/IP architecture. It is indeed noticeable for the case of this test when there is no reply at all, which could imply that the problem could be in the local part, or in the ISP network. Consequently, this checking will be performed at the right moment in the automated network diagnosis, implementation we will discuss in the next section.

In the event that the requested servers are not reachable, a traceroute [9] command could be executed to determine in which node the packets are not transmitted. Traceroute aims at determining the routing path between the endpoint source and a targeted host. Its process is based on sending packets, UDP, TCP or ICMP although UDP is the standard version, to the targeted host changing the IP protocol Time To Live (TTL) value of the sent packets. The purpose is to send a packet with a specific TTL value so that the next gateway in the path discards the packet and sends back an ICMP message "Time Exceeded" to the source. The process works in the same way, increasing the TTL at each step to determine the next router. However, the traceroute command may not provide any relevant information, because nowadays, the widespread use of firewalls block UDP and ICMP echo messages. Network administrators program their routers so that it does not answer to traceroute packets or even ping messages that are based on UDP and ICMP protocols. Another solution to trace the route could be to use another command "tcptraceroute" [10], based on sending TCP SYN packets. Admittedly, this method is quite unsure as we already tried to determine a route path to a host while pinging worked correctly, but the

[9] `http://linux.die.net/man/8/traceroute`

[10] `http://manpages.ubuntu.com/manpages/hardy/man1/tcptraceroute.1.html`

```
1  192.168.42.1 (192.168.42.1)  0.546 ms  0.327 ms  0.317 ms
2        * * *
 * * *
 * * *
```

Figure 23: Path trace to a public DNS server of Google, problem of routing in the access ISP network

procedure stopped at some point, for diverse reasons that could be firewalls, routers who do not respond to `traceroute` messages or simply that replies were lost or expired in the network.

An example of this command is shown as following in Figure 23. For instance, the user tries to ping the DNS server at the IP address 8.8.8.8, but fails, that is why a traceroute instruction is executed. We can observe that from the node 2, the device does not receive any reply after sending the UDP packets, even from the node 3 and so forth. Many reasons are possible, like the fact that the replies from the node 2 and all nodes beyond expired during the transmission because of wrong settings in the TTL values for instance or because it was lost. However, we would more consider that it also may be due to either a problem in node 1 that cannot route the packets, or that a firewall blocks and discards every packet at the node 2, which implies that the next hops cannot be reached, especially the targeted host. One solution could be at least to capture the received ICMP packets of the traceroute instruction. Indeed, the node 1 may not reach the node 2, because of a routing problem or a firewall. The node 1 will send a specific ICMP message with the Type 3 "Destination not reachable" with a specific code that could be in the range [0 –15], detailing the reasons.

Actually, this `traceroute` instruction may be used to identify ***Internet routing*** class, as the public DNS server is a public destination in the Internet. If the connection in the access ISP network is up, it may not be the case for the routing paths connecting the ISP network and the targeted public DNS server. The Figure 24 presents the results from the same instruction to trace the routing path between the endpoint source and the public DNS server at 8.8.8.8 in the event that the problem of routing occurs out of the access ISP network at the node 13. In fact, this case is identical to ISP routing example in Figure 23. It can be because the node 12 cannot route the packets to the node 13, or the node 13 is a firewall discarding every packet passing through. However, compared to ISP routing, the difference is that the node 13 belongs to another ISP, and the access ISP network is not to be blamed.

DNS Test Actually, referring to section 2.4.2, the DNS problems that could correspond to the fact that DNS servers of the access ISP may be down can be checked by using a DNS resolution-related tool that makes explicit DNS requests to resolve a hostname into an IP address. Such tools can be simple commands as

```
 1  192.168.42.1 (192.168.42.1)  0.546 ms  0.327 ms  0.317 ms
 2  host-94-101-2-142.igua.fi (94.101.2.142)  11.525 ms  8.234 ms  11.596 ms
 3  212.73.248.1 (212.73.248.1)  21.505 ms  8.591 ms  9.599 ms
 4  ae-4-8.bar1.copenhagen1.level3.net (4.69.148.166)  30.911 ms  33.524 ms  34.217 ms
 5  ae-7-7.ebr1.dusseldorf1.level3.net (4.69.142.170)  50.777 ms  45.975 ms  48.620 ms
 6  ae-45-45.ebr3.frankfurt1.level3.net (4.69.143.166)  49.502 ms
   ae-47-47.ebr3.frankfurt1.level3.net (4.69.143.174)  49.403 ms
   ae-48-48.ebr3.frankfurt1.level3.net (4.69.143.178)  59.711 ms
 7  ae-63-63.csw1.frankfurt1.level3.net (4.69.163.2)  48.611 ms
   ae-83-83.csw3.frankfurt1.level3.net (4.69.163.10)  50.016 ms  49.190 ms
 8  * ae-3-80.edge3.frankfurt1.level3.net (4.69.154.135)  53.279 ms *
 9  google-inc.edge3.frankfurt1.level3.net (212.162.24.18)  49.098 ms  78.943 ms
   google-inc.edge3.frankfurt1.level3.net (212.162.24.14)  59.931 ms
10  209.85.248.12 (209.85.248.12)  49.831 ms  72.507 ms
    209.85.254.108 (209.85.254.108)  51.848 ms
11  209.85.251.178 (209.85.251.178)  44.816 ms
    209.85.251.180 (209.85.251.180)  44.227 ms
    209.85.251.246 (209.85.251.246)  49.547 ms
12  209.85.254.114 (209.85.254.114)  49.618 ms  44.068 ms
    209.85.254.112 (209.85.254.112)  44.600 ms
13  * * *
  * * *
```

Figure 24: Path trace to a public DNS server of Google, problem of routing in the Internet

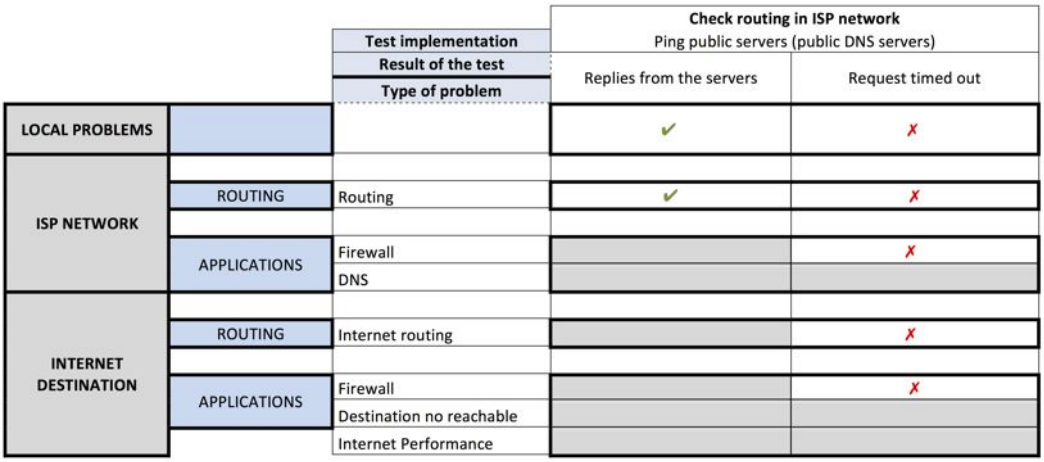


Figure 25: Test for ISP routing related problems

textttnslookup[11] that is usable Windows. However, this command has a different DNS resolving method, that is why it is more recommended to use `dig`[12] or `host`[13] commands that are available on UNIX for instance.

An important notice to recall is that the `ping` command is not suitable for DNS testing as it does not use the same method to map DNS names to IP addresses[14]. The mapping is indeed fetched at first in the file `host`, which may lead to unusual results compared to `nslookup`, `dig`, `host` exclusively relying on DNS operating.

With two of these commands, the test consists in resolving a hostname of a webserver for instance with the following instructions and compared the results:

```
$ host -t A www.mywebsite.fi
```

```
$ nslookup
> set q=A
> www.mywebsite.fi
```

The instructions above aim at getting the A records that are the mapping between the DNS name www.mywebsite.fi and the IP addresses of the computers sharing the same name. Admittedly, a webserver may be often requested, that is why there are actually several computers with different IP addresses that share a same dns name www.mywebsite.com for instance. In the DNS Resource Records of the authoritative DNS server, different A records map the name www.mywebsite.com with different IP addresses. Then the authoritative DNS server does not return the same IP address all the time for every user, in order to avoid the device to be overflooded. This is called Load balancing. Then comparing the results from both instructions should lead to find common results if DNS process works properly.

In the event that we do no get any reply from DNS requests, the next step to check is to verify the IP addresses of the DNS servers that were provided with IP

[11] `http://linux.die.net/man/1/nslookup`

[12] `http://linux.die.net/man/1/dig`

[13] `http://linux.die.net/man/1/host`

[14] `http://homepage.ntlworld.com/jonathan.deboynepollard/FGA/nslookup-results-different-to-ping.html`

configuration during DHCP process. These IP addresses can be found in the file `/etc/resolv.conf` in UNIX OS. For Windows, the command `ipconfig` displays the main elements of IP configuration, among which the IP addresses of the DNS servers. The objective is to ping these IP addresses and to confirm that the servers are up. It is possible to also scan the server ports to confirm that it is a DNS server by using the `nmap` command [15], which can be used on UNIX and Windows OS. It aims at finding which TCP ports of a device are open to determine which type of services the device provides [27].

```
$ nmap -Pn 130.233.224.132

Starting Nmap 6.40 ( http://nmap.org ) at 2013-09-11 15:51
EEST
Nmap scan report for ns01.aalto.fi (130.233.224.132)
Host is up (0.0023s latency).
Not shown:  999 filtered ports

PORT STATE SERVICE
53/tcp open domain
```

These lines show a port scanning on a device that is a DNS server (whose IP address is 130.233.224.132), as we can observe with the TCP port 53. If the DNS server seems to be down, the last process is to resolve a DNS name using a public DNS server, after verifying that the public DNS server is up. For instance, the public DNS server of Google can be requested to do so by one of these instructions:

- `nslookup www.mywebsite.com [public dns server or ip address]`
- `host -v -t a www.mywebsite.com [public dns server or ip address]`
- `dig [@public dns server or ip address] www.mywebsite.com`

If this process succeeds, then the problem may come from the local DNS server provided by the access ISP, which for unknown reasons cannot perform DNS requests. If the request cannot lead to a DNS resolving, then we can assume, the problem comes from the external process of resolving performed by the authoritative and non-authoritative dns servers requested by the public dns server.

The Figure 26 sums up the main result of what a DNS checking should inform to the user. The instructions are not specified, this table just helps to define which kind of instructions are processed to target the main problem DNS: the previous paragraphs defined with more explanations what the detailed instructions are to isolate one of the subclasses of DNS problem: Local DNS server is down, external DNS resolving process does not work, etc.

[15] `http://nmap.org/`

		Test implementation	DNS checking Commands for DNS queries: nslookup, host, dig	
		Result of the test	Correct reply	Wrong/No reply
		Type of problem		
LOCAL PROBLEMS			✔	✗
ISP NETWORK	ROUTING	Routing	✔	✗
	APPLICATIONS	Firewall		✗
		DNS	✔	✗
INTERNET DESTINATION	ROUTING	Internet routing		✗
	APPLICATIONS	Firewall		
		Destination no reachable		
		Internet Performance		

Figure 26: Test for DNS related problems

Destination Checking The Internet connection problem could prove to be simply the fact that the destination host the user would like to establish a connection may be down. A simple test to check this possibility is to ping the destination host. However, pinging directly the host by the hostname should not be relevant, as the malfunction could come from DNS services. In this case, the DNS checking should be performed before testing the reachability of the destination server. What we suggest for this *Destination Checking* is to launch a process to solve the hostname of the destination at first, using the DNS solver commands mentioned in the *DNS checking* part, and if DNS works properly, pinging one of the set of IP addresses provided.

As we can observe once again on Figure 27, it is not really relevant to perform this test at first in the event that we do not get any reply from the destination as there are too many uncertainties to determine which is the class that is the trouble source.

As a matter of fact, this instruction can be useful to check also the *Internet routing* class by analysing the ICMP messages received after trying to ping the destination host. In the event that the destination is not reachable, it may be possible to receive an ICMP message type 3 (Destination unreachable) from the last router that tried to route the packets to the next hop, which is not necessarily the targeted host. It would imply that there is a problem of routing in the Internet, between the access ISP network and the targeted destination. Consequently, another test can be to trace the routing path from the endpoint source to the destination and analyse the results as in Figure 24.

3.3.4 WLANS: List of tests

This subsection deals with the tests for WLANs.

Different classes from wired LANs to take into account As we already highlighted this point in the section 3.3.2, due to differences in the architecture of

		Test implementation	Destination checking: is it reachable? Ping Destination IP address after checking that DNS works	
		Result of the test	Reply from destination with good RTT and packet loss rate	No reply/Bad RTT, packet loss occurs
		Type of problem		
LOCAL PROBLEMS			✔	✗
ISP NETWORK	ROUTING	Routing	✔	✗
	APPLICATIONS	Firewall		✗
		DNS		
INTERNET DESTINATION	ROUTING	Internet routing	✔	✗
	APPLICATIONS	Firewall		✗
		Destination no reachable	✔	✗
		Internet Performance	✔	✗

Figure 27: Test to check if the destination is up/down

wired LANs and WLANs, some additional classes of problems must be taken into account for WLAN troubleshooting. This section deals with the additional specific tests for WLANs, but obviously, the common tests for wired LANs and WLANs are included in the WLAN procedure for the automated system.

The Figure 28 sums up all the considered classes of problems related to WLAN. We displayed the table for *Coverage checking*, explained in the next paragraph *No coverage*. As we can observe, most of the classes are common to wired LANs thanks to OSI-layered model, but some are exclusively specific to WLANs, particularly in physical layer, and network access.

No coverage Some software already exist for wireless network troubleshooting such as wireless networks scanner (*Wi-Fi Explorer* ([25]) for MAC OS, *inSSIDer* ([3]) for Windows). These tools aim at scanning the available wireless networks around the mobile station. The coverage checking can relies on this method by measuring the Signal to Noise ratio of all the surrounding access points of a specific wireless network (identified by its Service Set Identifier (ESSID)) to verify if there is enough signal to connect to an access point belonging to the wireless network the user would like to get access to.

What we can expect from this process is to get information about the level of signal:

- "Signal is good": The quality is good enough so that the mobile station can connect to the access point.
- "Low Signal": The access point is up, but the best signal is not enough high to connect to the network.
- "No signal": The wireless scanner could not find signal at all. The access points may be down, or the cause could be inside the mobile station which is not able to get a signal because of internal problems like NIC malfunction.

		Test implementation / Result of the test / Type of problem	Test Coverage: Signal measurements — The signal is good	Low signal	No signal
LOCAL PROBLEMS	TCP/IP Stack implementation	Wrong installation: reset TCP-IP stack			
	PHYSICAL LAYER, LINK LAYER	No NIC/NIC broken	✔	✔	✗
		No coverage	✔	✗	✗
		First-hop broken			
	NETWORK ACESS: authentication	Wrong credentials			
		Authentication server			
	IP LAYER: No IP address	Wrong settings to get IP address, IP configuration			
		DHCP			
	LOCAL "Routing"	Gateway problem			
		Authentication Web			
ISP NETWORK					
	ROUTING	Routing			
	APPLICATIONS	Firewall			
		DNS			
INTERNET DESTINATION					
	ROUTING	Internet routing			
	APPLICATIONS	Firewall			
		Destination no reachable			
		Internet Performance			

Figure 28: Test to check coverage quality

Authentication Web This class is misleading since the client device which connects to wireless networks secured receives a proper IP configuration to transmit data packets over the Internet. However, the traffic is blocked and the client is redirected to an authentication process. Actually, there are different methods used in a captive portal to automatically redirect the user to the authentication portal:

- HTTP redirection: this is the most common method. After connecting to the network, the client opens a web browser, enters an URL to browse a website. The URL is resolved into IP address by the DNS server of the local network, and the web browser can send an HTTP request to the webserver to get the webpage. Then, the authenticator in the network blocks the request and sends back an HTTP message containing a redirection link to the login webpage for authentication. This redirection HTTP message has the 302 Code.

- IP redirection: Considering the same example of browsing a webpage by entering the URL in the web browser, a DNS query is sent to the DNS server that resolves it into the actual IP address. The client sends packets with this IP address but the gateway alters the IP address by replacing it with the IP address of the Captive Portal.

- DNS poisoning: The IP configuration that the client receives after establishing

a connection is only intended for unauthenticated users. In this case, the DNS server resolves the URL into the IP address of the Captive Portal. The client makes a request to this IP address and gets the Captive portal login page.

We decided to study HTTP redirection in particular. We made a small experiment to confirm our assumptions about the fact that some possible short tests are not sufficient to identify this specific problem. In a restaurant, we tested this type of wireless network. User id and password were not required, but the access was granted after agreeing to the terms of use displayed on a webpage sent by the authenticator. We did not agree on a webpage but we observed that our network interface had a proper IP configuration, i.e an IP address, the IP address of the default gateway, and also the IP addresses of the DNS servers. Pinging the gateway was possible although some packets were lost, and querying the DNS server for DNS resolving worked too, as shown on the packet capture in Figure 30. We sent different DNS queries for different websites using `nslookup` and `host`. We compared the results with the same queries we performed on another network and we observed that the IP addresses translated were identical, which confirmed that the redirection was not based on DNS poisoning.
At this stage, a user can think that he can use Internet services, because the local link works as a matter of fact. However, when we tried to ping public dns servers, for instance Google's (whose IP address is 8.8.8.8), the ping request times out. It implies that the default gateway does not route the ping packets. It could have been due to a problem of routing, but in this case, the network access is blocked by authentication web.

To identify this problem, a possible method is to make the client sends an HTTP request to get a webpage and then capture HTTP packets with a traffic analyser to examine the characteristic of the HTTP traffic sent by the webserver/authenticator to confirm that this authentication method is related to captive portal with HTTP redirection. Actually, some operating systems process this request automatically, like iOS from Apple by fetching the webpage at the URL at [16] [24]. We captured HTTP traffic and filtered HTTP messages with the code "302 Redirect". We found indeed a corresponding HTTP message showed on Figure 30. We can read in the Line-based text data that there are XML tags `<LoginURL>` and `<WISPAccessGatewayParam>` referring to a redirect link. Admittedly, the names of the XML tags change depending on the network, but this proves that the HTTP message is the one enabling the HTTP redirection to the captive portal. It can be a method to check if there is a captive portal.

Nevertheless, as a packet analyser may present some aspects of private use problems, a combination of tests can be done instead to proceed by process of elimination of classes, which are in this case, as we can refer to Figures 16 and 25 in WLAN part: if the IP configuration is correct, but there is something wrong in ISP routing, it can be due to a problem of Authentication Web.

[16] `http://www.apple.com/library/test/success.html`

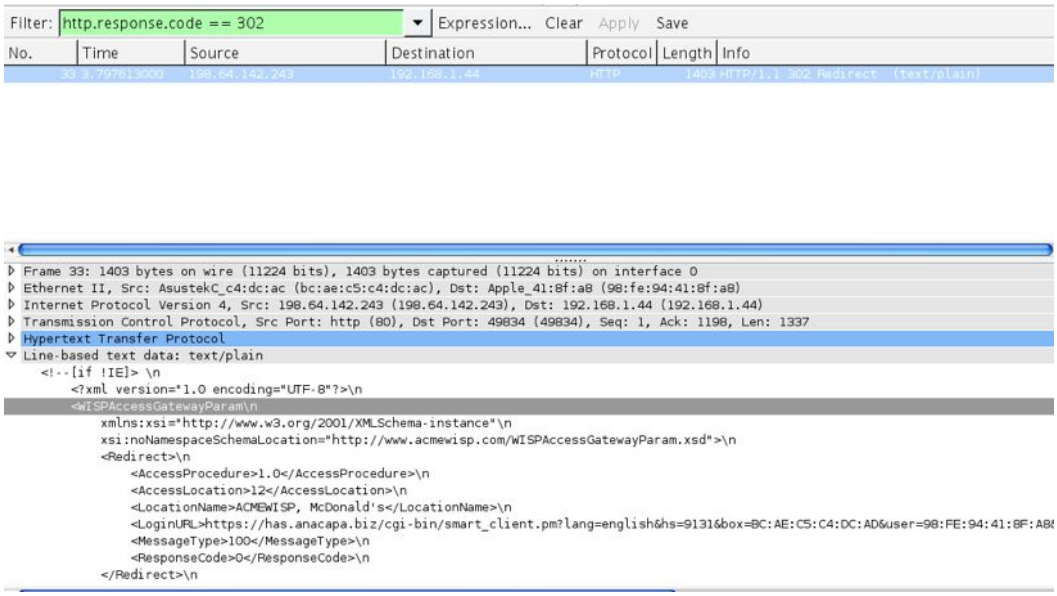


Figure 29: Capture of HTTP 302 Redirect messages for Captive Portal

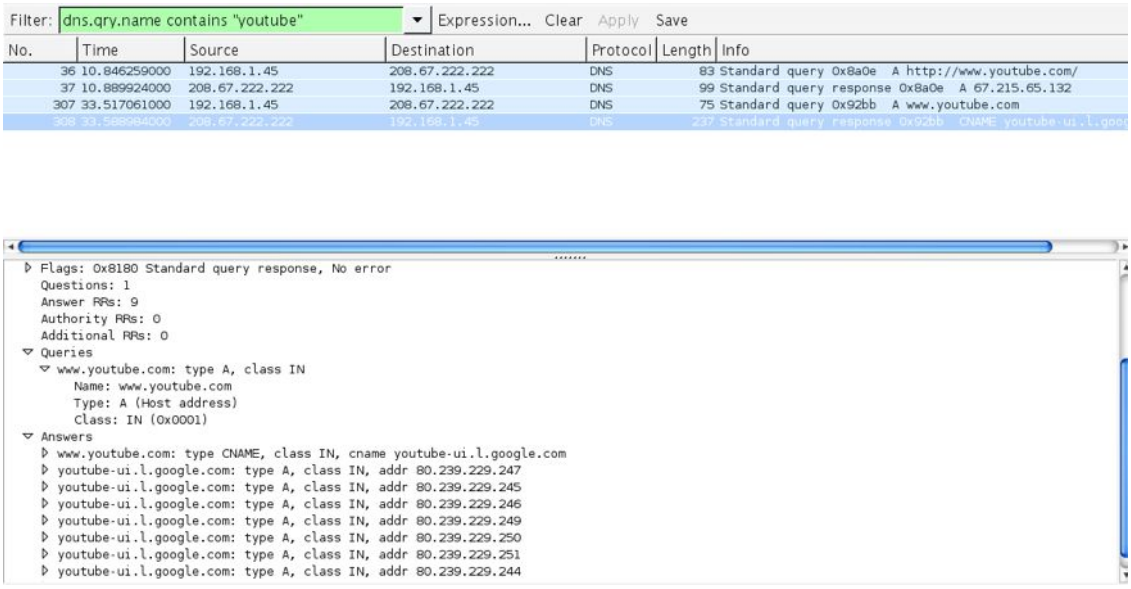


Figure 30: DNS query for resolving "www.youtube.com"

ISP Routing Checking The test designed for wired LANs is also usable for WLANs as the protocol layers involved are common: the results from the test on Figure 25 (wired LAN part) are interpreted in the same way as the wired LANs. However, we made a comment in the wired LAN part about a gateway problem that is more characteristic to WLANs. Indeed, the device can be misled and connects to a wireless network set by a printer. The IP configuration seems correct: the device gets an IP address, a default gateway IP address but when it tries to fetch a webpage, there is no reply: the gateway is actually the printer.

In the event of a wired LAN, after noting that none of the requested public DNS servers replied back, the automated system would end the procedure confirming that the problem may regard ISP routing to the user. For WLAN case, there is an additional test to check that the gateway is really a router and not a printer or another device. This can be done by scanning ports of the fake gateway, using again the command "nmap" [17]. An example of how this command solved case is explained as following:
The fake gateway whose IP address is 192.168.0.1 for instance replies back to ping requests. To scan its ports, we launch the command:

```
$nmap -nr 192.168.0.1

Starting Nmap 6.40 ( http://nmap.org ) at 2013-09-10 15:38
EEST
Nmap scan report for 192.168.0.1
Host is up (0.017s latency).
Not shown:  999 closed ports

PORT STATE SERVICE
515/tcp open printer
```

The port 515 matches printer services, confirming that the device at the IP address 192.168.0.1 is a printer and not a router, otherwise the command would have returned a result like "Host is up. All 1000 scanned ports are closed", which is normal as a router works in the IP layer. Consequently, the command `nmap` can be really useful but is also controversial as it is also used for malicious attacks.

3.3.5 Classes that can not be checked specifically

Looking through the tables on Figure 10 and 11, we note that some classes of problems are left without suitable tests, as it is due to their external characteristic. Actually, some tests are possible, although the information provided may be uncertain and thatthe tests could be compromising when used.

Internet Routing We did not propose an explicit test for checking the Internet routing and assuring that the problem comes for this very class, but actually, by

[17] `http://nmap.org/`

combining the tests that we explain in this chapter, assumptions can be made to tell whether the problem comes from this class or not. This will be shown in the next Chapter detailing the automated procedure. Indeed, referring to the tables about *ISP Routing* checking and *Destination* checking in Figures 25 and 27, we observe that failed pings to the targeted destination and the public DNS servers may be due also to a problem of routing in the Internet. Routing paths are evidently not the same depending on the destination host, which is why the fact that a routing path to a specific host is up does not imply that another host is also reachable in the Internet.

Firewalls Firewalls can be placed all over the Internet in ISP networks. As it blocks specific ports, applications and protocols, it is difficult to deduce where the problems come from. It may be a firewall that is indeed blocking the specific traffic, or just the requested host for the specific protocol or port that does not reply. We could classify firewalls in two categories:

- The local firewall that is at the border between the local network and the access ISP.
- Firewalls located further in the network, in the public Internet, out of the access ISP network.

The main issue of firewalls is that the common configuration of firewalls is done in a way that firewalls silently discard packets that are not allowed. Consequently, a source host is not aware that its traffic is blocked by a firewall, but can only assume that the destination host is unreachable. In theory, a simple check could be to send different types of packets based on different protocols and requests: different websites, DNS, ICMP echo, at first having checked that the local network works. There are some methods explained to check the presence of firewalls. One simple test is to perform a traceroute to get the path to the targeted host, with the `traceroute` command, like the example we mentioned for *ISP Routing* in Figure 24. Noting that from the node 13, packets are not transmitted. There is a method called "Firewalking" [18] which consists in detecting a firewall and perform a port scanning to determine what the firewall does not allow. With our example, it consists in at first determining the IP address of the node next after the firewall, i.e the node 14 in our example, by sending UDP port 53 packets corresponding to DNS and that is not blocked by firewalls. These UDP port 53 packets are sent with the next TTL value, i.e 14. The packets will go through the node 13 and will exceed at the node 14, which will reply back with an ICMP message "Time Exceeded" and informing about its IP address. Consequently, it will be possible to scan ports of the node 14 to check which ports and protocols are blocked by the firewall at node 13. Malicious users perform ports scanning and banner grabbing[19] to determine the ACL filters of the firewalls to find the vulnerabilities of hosts. For this purpose, they use network

[18] `http://www.giac.org/paper/gsec/312/firewalk-attackers-firewall/100588`
[19] `http://fabian-affolter.ch/blog/banner-grabbing/`

tools such as `nmap`, `netcat`[20], `telnet`[21], etc. Obviously, these methods are not legal, that is why we choose not to use port scanning in this class and restrict our test to a simple traceroute instruction.

Internet performance Admittedly, this class is part of the problems we should take into account. A simple test to measure Round Trip Time and packet loss rate can be carried out, but it will just roughly inform the user about the quality of Internet connection like Mean Opinion Score (MOS). As many settings are concerned, Quality of Service is a complex issue that must be considered separately from no-connectivity, the main topic of this thesis.

3.4 Cellular mobile networks

Admittedly, one important point to notice is that in most of cases, the user equipment for cellular networks are smartphones, PDA, tablets, which are devices with capacities more limited than computer devices. In theory, the proposed following tests would be feasible on a cellular network from a laptop connected to a GPRS or UMTS Terminal in order to process the protocols of the physical layer, link layer and network layer of the cellular network, but implementing it for concrete purpose is another issue due to limited capacities of smartphones. Consequently, we consider that the tests in cellular networks are processed by a computer, and not a smartphone.

3.4.1 List of tests common to LANs

In this section, the suggested tests are common to LANs and have already been explained in Section 3.3. Even though the involved technologies are different, and that tests must be tuned to cellular network protocols, the principles of the tests remain unchanged, compared to LANs, and the results from the tests lead to the equivalent conclusions.

- ***TCP/IP software checking***: As it regards protocols in the IP layer and above, this test is exactly identical as LANs TCP/IP software checking.
- ***Network Interface Controller checking***: Although the network adapter does not rely on identical technologies as LANs, the checking is processed with the same objective: aiming at checking the status of the network adapter relying on NIC driver implementation, and depending on the result, would inform the user that the network adapter is the cause of Internet no-connectivity.
- ***Coverage checking***: As *Coverage checking* in WLANs, it consists in measuring the signal level from the Base Stations or the Nodes B, depending on the type of cellular network the user equipment wants to connect to.

[20] `http://linux.die.net/man/1/nc`
[21] `http://linux.die.net/man/1/telnet`

- ***IP Configuration checking***
- ***Authentication checking***: In LANs, we suggested to use a packet analyser to capture the packets to analyse a potential authentication running. This test could be tuned for cellular technologies in the exact same way, as packets analyser specific to cellular networks already exist [22]. Authentication is based on the International Mobile Subscriber Identity, stored in the SIM card, but GGSN performs also ISP authentication based on RADIUS access. So in theory, it is feasible to capture packets and examine the authentication process performed by the SGSN and GGSN, but this packet capturing may be confronted with private usage issues, as already pointed out. In LANs, an alternative solution was proposed, based on processing by steps of elimination: if the components of the layers lower than authentication work, but there is no IP configuration, authentication must be the cause.
- ***Gateway checking***: The gateway which is in this case the GGSN can be requested by `ping` command, carrying out the same test as LANs. Nevertheless, some ISPs can also block ICMP echo messages, so that the GGSN does not reply. Consequently, this possibility must be taken into account for the results of this test regarding cellular network.
- ***ISP Routing Checking***: Again, as LANs, it can also be performed using ICMP echo messages to public webservers by its IP addresses.
- ***DNS Checking***: There is no change for this test as it can be directly performed over the cellular network using a computer connected with a cellular terminal. However, carrying out such a test on a mobile phone is another issue that we will discuss in the next section.
- ***Destination Checking***

3.4.2 Classes that can not be checked

As we already noticed that cellular troubleshooting is more meant for smartphones, a certain number of classes tests must be dropped, because of the limited resources of the mobile devices. Moreover, cellular networks can be more protected against some protocols such as ICMP echo, which may lead some of our tests to insignificant information and a waste of energy.

First-Hop Checking In LANs case, we proposed a test to verify if the no Internet-connectivity was due to the *First-Hop broken* class by examining ARP table in the user equipment. However, ARP protocol is not used in cellular networks, that is why this test can not be processed in cellular networks. Actually, we also suggested to capture all packets on the network interface to confirm that the link is on, even though the user device may not be allowed to carry out Internet protocols. Traffic

[22] `http://www.gl.com/umtsanalyzer.html`

analysers for mobile cellular networks intended for computer devices already exist, which means that it could be designed for a smartphone. Nevertheless, the main issue is that a traffic sniffer is not allowed on a smartphone, then this method cannot be used on such a device. Consequently, we dropped both mentioned tests. The test that could be roughly equivalent to *First-Hop* Checking is the *Coverage checking* that would be explained in the next parts, but admittedly it is just a physical link-related checking, compared to the others that tested the logical link layer.

DHCP Checking In UMTS, IP address assignment is performed by the GGSN. We could trace packets from GGSN to examine DHCP protocol like in LANs, but in the end, what the user is interested in is the fact that he does not have a proper IP configuration, and the cause is external as it is ISP responsibility to fix it.

3.4.3 List of tests specific to cellular networks

No SIM Card The test simply verifies that the mobile device is equipped with a SIM Card that guarantees the access to the cellular network, and in this case Internet access if this service is part from the subscriber agreement. The results returned by the test are:

- "SIM card OK": The SIM card works
- "No SIM card": The SIM card may be absent or broken.

Data Limit Actually, this class coincides with *Authentication problem*, and is also difficult to be pointed out, because it is related to ISP policies regarding network access. If the subscriber is not allowed for a period of time to get access to the network, for different reasons such as roaming, the ISP blocks the requests of the mobile device by methods we are not sure of, but we can assume that it is done through the subscriber authentication with the IMSI.

Consequently, this class can not be verified with an explicit test, but by proceeding by process of elimination, we can suggest that by relying on facts that all components in lower layers work (physical, link and network), the problem may be located in an authentication class.

4 Application to show to the user

In the previous chapter, we presented the theoretical tests to target each main class of problems mentioned in Figures 10 and 11. However, we decided that some classes displayed in these tables will not be checked because of their complexity or because the proposed tests could pose a problem of legal use or could not return significant results due to protections and firewalls in the network.
This chapter focuses on implementing the automated procedure in order to proceed the tests that have been presented previously.

Distinct factors are to be considered Before executing the automated procedure, the automated system has to deal with some parameters which are the type of user equipment and the type of network connection. In this work, we decided to take into account wired LANs, WLANs and mobile cellular network. Regarding the user equipment, we will implement the automated procedure for a computer device, and not a smartphone. As a matter of fact, most of the tests that were explained previously pertain to UNIX commands, among which some are also usable on Windows OS. Theoretically, these application commands can be implemented in an operating system intended for smartphone such as Android [23], which is Linux-based. This characteristic facilitates the adaptation of tests meant for computer, but it is not impossible to make it suitable on other mobile operating systems. Indeed, these application commands such as *ping*, *nslookup* or *tcpdump* are based on the application layer protocols of the TCP/IP model.

Moreover, the main issues for smartphones are the restricted capacities of the mobile devices, i.e the battery and the Central Processing Unit amongst others. Another issue concerns traffic analyser that we mentioned for several tests such as *First-Hop*, *DHCP* and *Authentication* which are not allowed on smartphones. The tests based on capturing traffic cannot be used for a smartphone automated tool, which leads to drop these specific tests in WLAN and mobile cellular network checking from a smartphone. For these reasons, a developer who would like to implement such an automated tool would have to adapt the tests and instructions and limit the traffic generated by the checking procedures in order to save mobile resources.

4.1 Necessity to combine and order the tests

4.1.1 Combining the tests

In Chapter 3, we already noted that most of the tests that we proposed and described to target a class of problems do not provide significant information if the automated system performs each one alone, independently from the other tests. This is rather well illustrated in the table displaying the results for the *ISP routing* test in Figure 25. Only carrying out this test to check the network connection may lead to several uncertainties in the event that the ping did not return any reply from the public

[23] `http://www.android.com/`

server. Indeed, as we can observe, the problem may be in *Routing* class or in *Firewall* classes or in any class of the *Local problems* category.

Moreover, the table does not display another case of result which is also important to take into account. As a matter of fact, the ping may return an error message, or a strange value for IP address. The cause can be an incorrect configuration of TCP/IP software, but for a usual user, he may not know how to interpret this error message. Consequently, this note is essential because it shows that even the tools the tests rely on can also be flawed. This is why we already pointed out that the first step of the automated procedure is to check that the tools we will use for the tests, i.e network-based application commands, are properly set and function.

4.1.2 Find an efficient ordering

Next issue is to define an optimized and efficient order to carry out the tests to quickly target the problem. The principle of TCP/IP architecture is useful for taking this decision. One option could be to check at first the upper layers of TCP/IP and look through the stack from the top to the bottom. For instance, what people usually do is to ping a well-known webserver. In the event that the requested host replies back, many checking procedures will be avoided. Indeed, we can deduce from the results that the problem may be in a higher layer than the network layer, for instance in *DNS*, *Firewalls* or simply that the requested Internet destination is down. However, if the `ping` instruction leads to no reply, then the problem may be in any class lower than ICMP echo protocol, except Firewalls that can also block ICMP messages. Then the next test must be performed in the lower classes.

For the moment, we explained the list of classes and tests assuming that only one problem could be the cause in the Internet connection. Admittedly, one problem is enough to break down the Internet connectivity or at least the use of Internet-based applications, but it may be possible that more than one problem occurs at the same time, in different classes, in different layers. The Figure 31 refers to a situation in which two problems happen at the same time, in different classes. We can interpret the cells of the stack as TCP/IP layers as the classes of problems of the table in Figure 15, as we tried to classify it according to TCP/IP stack. The tables in Figure 32 display the information provided by tests. The results from each test are shown considering that each test was carried out alone and independently from the others. For instance, Test A, carried out alone tells that the layer 5 does not work, making us assume that a problem can occur in this layer or in the lower layers 1, 2, 3 or 4. Indeed, the layer 5 cannot function properly, because there is something wrong in lower layers.

From the top to the bottom We will explain now what happens when we proceed the tests from the top of the stack to the bottom. The whole procedure can be summed up in the Table 1.
Step 1: we start by the Test A at the top of the stack in Class 5 (or Layer 5), which does not work. For the reasons we previously explained, the Test A makes us assume that the classes (or layers) concerned by the problem are **{5, 4, 3, 2,**

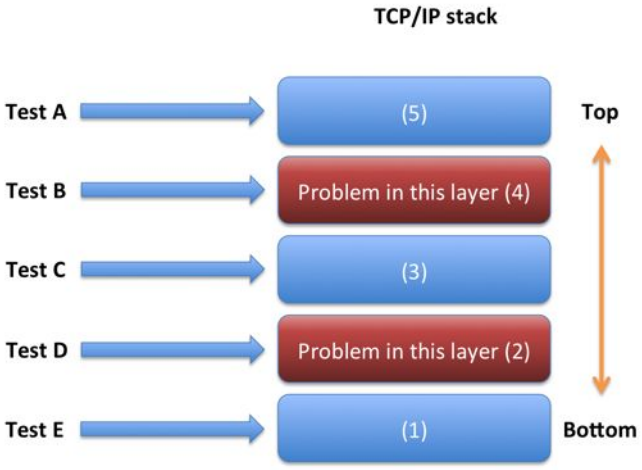


Figure 31: 2 problems at a same time, iteration 1

	Test A	Test B	Test C	Test D	Test E
5	✖	Unknown	Unknown	Unknown	Unknown
4	✖	✖	Unknown	Unknown	Unknown
3	✖	✖	✖	Unknown	Unknown
2	✖	✖	✖	✖	Unknown
1	✖	✖	✖	✖	✔

✖	The problem may be in the layer
✔	The entities and protocols work properly in the layer
Unknown	No information about the layer

Figure 32: Results of the tests corresponding to iteration 1

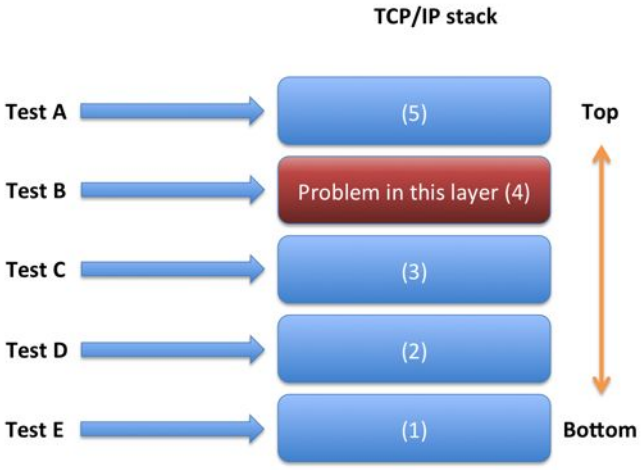


Figure 33: 2 problems at a same time, iteration 2

	Test A	Test B	Test C	Test D	Test E
5	✖	Unknown	Unknown	Unknown	Unknown
4	✖	✖	Unknown	Unknown	Unknown
3	✖	✖	✔	Unknown	Unknown
2	✖	✖	✔	✔	Unknown
1	✖	✖	✔	✔	✔

✖	The problem may be in the layer
✔	The entities and protocols work properly in the layer
Unknown	No information about the layer

Figure 34: Results of the tests corresponding, iteration 2

Step	**Test**	**Class checked**	**Does it work?**	**Hypothesis location**	**Class to still consider**	**Next classes to be tested**
1	A	5	No	{5, 4, 3, 2, 1}	{5}	{4, 3, 2, 1}
2	B	4	No	{4, 3, 2, 1}	{4}	{3, 2, 1}
3	C	3	No	{3, 2, 1}	{3}	{2, 1}
4	D	2	No	{2, 1}	{2}	{1}
5	E	1	Yes	{1}	–	–

Table 1: Tests procedure from the top of the stack to the bottom, iteration 1

1}, shown in the column *Hypothesis location*. The actual issue is that we cannot tell if the problem is in this very Layer 5 or below. As we cannot tell if it really concerns the Layer 5, we put aside this possibility (column *Class to still consider*). The possibility that the problem may not be in Class 5 will be confirmed by the next step. The set of the classes {4, 3, 2, 1} where the problem could be are displayed in the column *Next classes to be tested* and must be checked. For the next step (*Step 2*), the highest class of the set is to be checked, in this case, it is Layer 4. As a matter of fact, if Layer 4 works properly, it implies that all classes below Layer 4 work properly too, then the problem comes from Layer 5.

Step 2: Consequently, we execute the Test B in Layer 4, exactly lower than Layer 5. Observing the results from Figure 32, the Layer 4 does not work either, which implies that the problem may not be in Layer 5 but there is one problem at least in one of the set **{4, 3, 2, 1}**. We remove Layer 5 from the set *Class to still consider*, because there is at least one problem below, and update it with Class 4. Like Class 5, as we can not have much more information about Layer 4, we also put it aside in *Class to still consider*, because the next step will inform whether the problem comes from Class 4 or not. The current set of class to check is {3, 2, 1}

Step 3: Test C returns a result informing that the Class 3 does not work either, because of the problem in Class 2. We do not know but there is actually no problem in Class 3. We deduce that the problem may be in Class 3 or below, but not in Class 4. The current class to consider is Class 3, and the next classes to check are {2, 1}.

Step 4: Class 2 does not work because the problem is in this very class. The automated tool assumes that the problem does not come from Class 3 and may be in {2, 1}. It will check Class 1 in next step.

Step 5: Class 1 works. The procedure stops. The current *Class to still consider* before updating is Class 2 (from the row *Step 4*). We can deduce that the problem is in *Class 2*.

Now that the problem has been targeted in Class 2, we assume the problem can be fixed, but when the user tries again the Internet application, it still does not work. The new situation is shown on Figure 33. This is the 2nd iteration of the procedure to target the 2nd problem. We keep the method of looking through the stack from the top to the bottom. The main issue is that the first iteration of the procedure did not provide information on Classes 5, 4 and 3, so we must start at the highest layer, which is Class 5. The procedure goes on from the top to the bottom, until one class is found working correctly. The steps are the same as Table 1, except that at *Step 3*, Test C returns a result informing that Class 3 works, as Class 2 has been fixed. The process stops at *Step 3*, and the *Class to still consider* from previous step is Class 4. The problem is in Class 4.

With this method, the automated system checked Class 5 twice although Class 5 was working correctly. In the event that there is more than one problem, starting at the higher layer presents a main issue. It takes the risk to check a same higher Class several times although it does not have problems but does not work because of lower problems. As this procedure goes on until a Class is found to work properly, each iteration enables to find the lowest problem, and all the previous steps must

be performed again, which is not really efficient.

From the bottom to the top As a matter of fact, we propose to carry out the automated procedure by starting at the lowest layer of TCP/IP stack and look through it to the highest layer. The Table 2 shows both corresponding iterations to the example of two problems at the same time.

The first iteration leads to find that the problem occurs in Class 2.
Step 1: Results from Test E informs that Class 1 works. We deduce that the problem may be in {5, 4, 3, 2} (*Hypothesis location*), and we put aside temporary Class 1 as the highest class that works properly (*Highest Class working*). Consequently, the classes to be tested are {5, 4, 3, 2}.
Step 2: The next class to be tested is the lowest of the set *Next classes to be tested* from previous step, i.e Class 2. The Test D shows that Class 2 does not work. Consequently, the problem is in Class 2, and the procedure stops.

After fixing the problem in Class 2, the variable *Highest Class working* is updated with Class 2. However, the Internet access or Internet-based application still does not work. The procedure is again performed for a 2nd iteration, displayed in Table 2.
Step 1: Instead of checking Class 1 once again, the automated tool looks at the last recent value of *Highest class working* when the procedure stopped, which is Class 2. Consequently, it can starts the tests in the Class just above the *Highest Class working*, i.e Class 3, according to the table. Class 3 works, so the problem may be in the set {5, 4}, which must be tested.
Step 2: The lowest of the set, i.e Class 4 is tested, and it does not work. The procedure stops and Class 4 must be fixed. After being repaired, *Highest Class working* is updated with the value Class 4.
Then Internet access works properly, so it definitely ends the procedure. However, if it was not the case, a 3rd iteration would have been carried out, directly starting in Class 5.

In this way, we explained why we found this method more efficient, as we do not have to perform several times a test in a same layer. Actually, we can check all the layers through only one whole iteration just to make sure that all classes work. Consequently, we will apply this method for our automated procedure, explained in the next section.

Iteration, Step	**Test**	**Class checked**	**Does it work?**	**Hypothesis location**	**Highest Class working**	**Next classes to be tested**
1,1	E	1	Yes	{5, 4, 3, 2}	{1}	{5, 4, 3, 2}
1,2	D	2	No	–	–	–
2,1	C	3	Yes	{5, 4}	{3}	{5, 4}
2,2	B	4	No	–	–	–

Table 2: Tests procedure from the bottom of the stack to the top, iteration 1 & 2

4.2 Structure of the automated procedure

In this section, we analyse the global structure of the procedure that the automated system will carry out to detect a problem and find its location. We already explained the method "*Bottom to the Top*" the procedure will rely on. Classes of problems must be tested in a specific order. At first, we can gather these problems into groups which will consist in three macro steps. The global procedure will be common to LANs and mobile cellular networks, and is based on checking these three macro steps respecting the method and following the order:

1. Settings
2. Bits go through the network
    - Local Link
    - End-to-End/Routing
3. Applications

Admittedly, Macro-Step 1 *Settings* is an exception to the method *Bottom to the top*. Actually, we decided to check this main category of classes of problems at first for the reasons we will explain in the corresponding paragraphs. We summed up the global automated procedure classifying all the classes of problems in the three main macro steps, as shown on Figure 35. On the left side of the diagram, we pointed out the three main macro steps the classes pertain to. On the right side of the diagram, we recalled the three main environments of the classes of problems. Except the macro step *Settings*, the Figure highlights the method *Bottom to the top* the procedure is based on. We will now explain each macro step in more details in the following subsections.

Settings As we pointed out this fact, the macro step *Settings* is an exception to the method *Bottom to the top*. We could have carried out this macro step at the same level as IP layer, as it is part of it, but we chose to perform this checking at first in order to be more efficient and gain time. *Settings* macro step is based on checking TCP/IP software, to avoid misleading results from our tests but also particularly checking IP configuration. Indeed, a proper IP configuration implies that most of Classes of problems in the Local problems are not involved. In the event that the IP configuration is correct, the automated tool can skip an important part of *Local link* checking and directly checks from the *Local Routing* category, which improves efficiency.

Transmission of bits *Transmission of bits* macro step regards the classes in physical layer, link layer and IP layer. Indeed, before checking that applications work, putting aside their different services options, we must ensure that their basic traffic, i.e frames and IP datagrams that encapsulate their application data units, is transmitted:

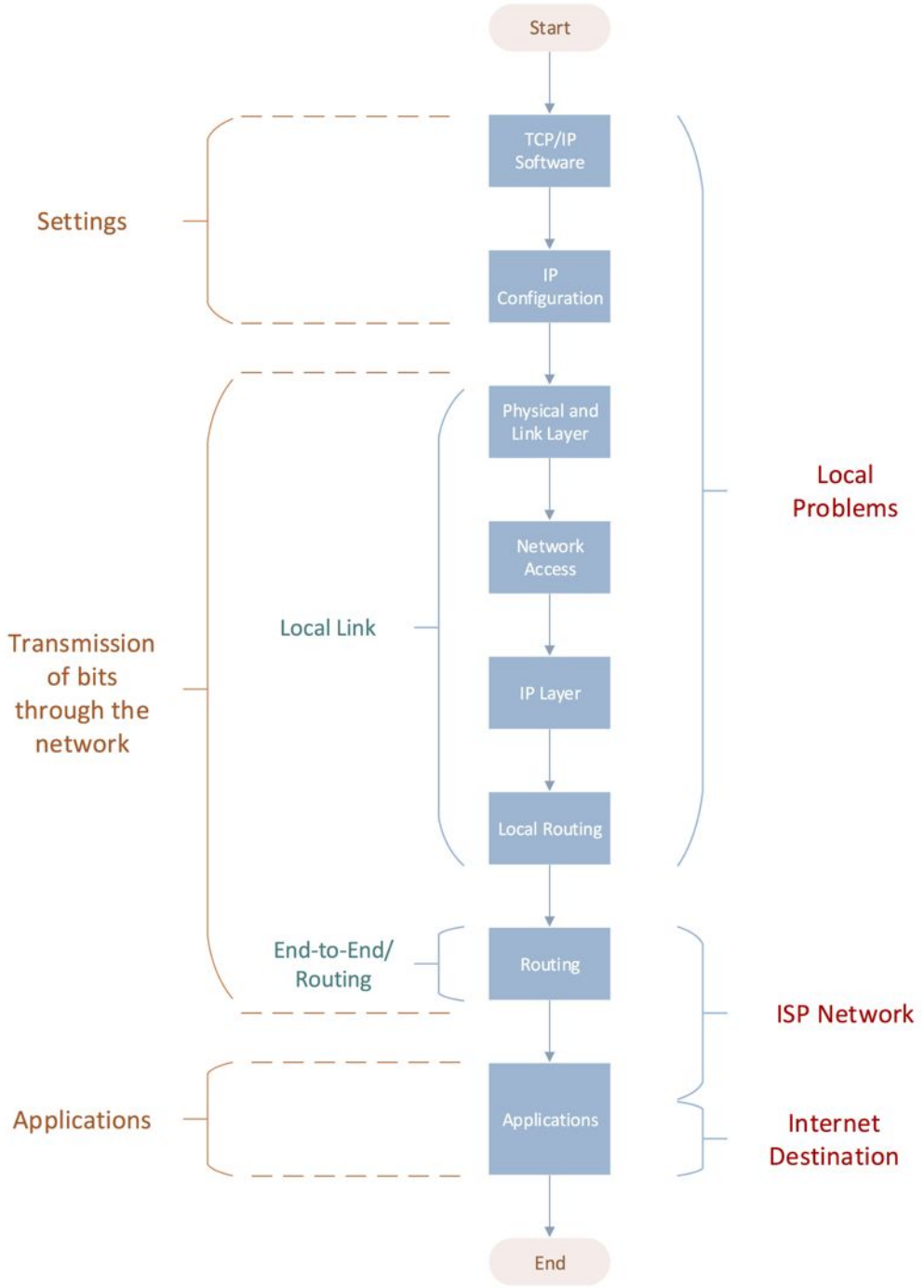


Figure 35: An overview of the automated procedure for network diagnosis

- Locally, from the endpoint to the Gateway, i.e the *Local link* which involves protocols from physical, link and network access layers, and also the Gateway (*Local routing*).
- And then outside, from the Gateway to the Internet destination, i.e *End-to-End* link or *Routing* through the access ISP network and the public Internet.

Applications After checking that the basic traffic could go through the local network and then through the public Internet, the macro step *Applications* checking can be carried out. It aims at targeting the problems in Applications services.

4.2.1 Presentation of the structure of the procedure

In order to present the functioning of the automated procedure, we relied on the method *Bottom to the top*, and we combined the tests based on the deductions provided in the tables in Chapter 3. From the combinations of tests, we illustrated the functioning with a state diagram whose steps are the different tests in Chapter 3. In order to make its reading easier, the whole state diagram is split into sub-diagrams, according to the three macro steps, which are also split into detailed steps, based on the tests. This subsection presents the different main global steps of checking (Figure 36) which are necessary to understand the order of the detailed sub-diagrams that would be explained in the next paragraphs. This global diagram is still common to wired and wireless LANs, and mobile cellular network.

Admittedly, the Figure 36 is an overview of the Figure 35, but the main point is the conditions imposed by the characteristics of the IP configuration from the macro step *Settings* as we already noted. Depending on the result from the *Settings* step, the next macro step will be either *Local Link (1)* or *Local Link (2)*. We represented a dotted green arrow from *Local Link (1)* to *Local Link (2)* to highlight the fact that passing from *Local Link (1)* step to *Local link (2)* can be done when all problems have been solved in *Local Link (1)*.

About the way some tests should be implemented for real The next sub-sections will deal with detailed sub-diagrams specific to each macro step, and for each type of network. We would like to add a few comments on the tests that are used for the procedure. Regarding the packets capturing aiming at targeting problems in First-Hop, Authentication, and other Internet applications, we defined that the fact of capturing packets from a specific protocol consists actually in capturing the whole packets traffic on the network interface from the moment when the user devices tries to establish a connection to the network. A new packet capture should be executed as soon as the user tries to establish a network connection once again. In order to analyse a specific protocol, the tool will apply specific suitable filters to isolate the traffic to be analysed to check the class of problem.

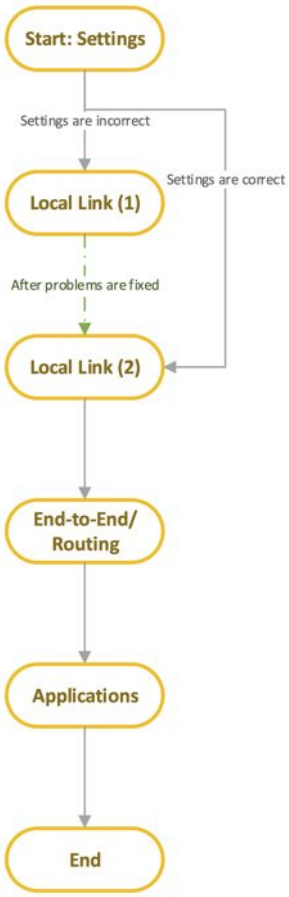


Figure 36: An overview of the global automated procedure for network diagnosis

4.3 Wired LAN and WLAN

This section deals with the sub-diagrams intended for Wired LANs and WLANs. As both networks present many similarities in the classes of problems, we chose to represent the checking procedure for both networks in a same sub-diagram for each macro step. As for mobile cellular networks, each sub-diagram is one macro step from the Figure 36 and is based on the tests from Chapter 3, with the corresponding results shown in their table. We tried to make captions coherent for all sub-diagrams.

4.3.1 How to interpret the state diagrams

As a matter of fact, every class of problems that we mentioned in wired LANs is common to WLANs. On each sub-diagram, specific checking steps intended for WLANs are displayed in purple color, which implies that for wired LANs, stages in purple are skipped.

The diamond blocks illustrate a test, the tool deduces from its results what the status of the checked class is, i.e the class does work or does not (grey boxes) and a next test is carried out. Boxes filled in orange or with a thick orange frame represent an event when a class of problem is suggested to be the cause to the user. At this stage, the procedure stops. Something must be done to fix and check every mentioned class more carefully even though there may be uncertainty about determining which is the real cause of trouble between two suggested classes. Actually, exceptions could be made, depending on the category where the stage is carried out. As a matter of fact, the procedure should stop after at least one problem has been found out because in most of the cases, the class is targeted with certainty, and without its good functioning, next tests cannot be performed properly. Nevertheless, exceptions could concern stages performed in Applications macro step, because the basic traffic of frames and IP datagrams would have already been checked and would work, which would not trouble applications tests, regarding a problem of basic traffic.

Moreover, we made a reference to the fact that the method *Bottom to the top* was interesting due to the following characteristic. Indeed, in the event that there is more than one problem, after suggesting a possible problem, the procedure stops but can start directly from the Class just above the identified Class of problem after the problem has been fixed. This characteristic is represented on all sub-diagrams with green dotted arrows, connecting two checking stages (i.e from one orange box to another state). It signifies that passing from one stage to the other can be done only after the problem has been fixed. Otherwise the procedure stops at the step when the automated tool suggests different classes of problems that could be the cause of no Internet connection. After all classes suggested by the tool are fixed and work properly, the procedure can go on, in case there is a another problem above in the TCP/IP stack.

4.3.2 Presentation of LAN/WLAN-related sub-diagrams

We present in this part the different sub-diagrams that form the whole global automated procedure for network diagnosis in wired LAN/WLANs. In the event that a sub-diagram was too unreadable because of too many stages, we used another sub-diagram to represent a succession of sub-tests. It is the case for DNS checking, which implied several short tests. The following list sums up the link between macro stages displayed for each stage diagram.

- Figure 37: Start from *Settings* to *Local Link (1)*. The tested classes pertain to the *Local Problems* category, especially TCP/IP software and IP settings.
- Figure 38: from *Local Link (1)* to *Local Link (2)*, for classes in *Local problems* category, with the conditioning that the IP configuration is incorrect.
- Figure 39: from *Local Link (2)* to *End-to-End/Routing*, for classes in *Local problems* and *(access) ISP network* category.
- Figure 40: from *End-to-End/Routing* to *Applications*, for classes in *Local problems* and *(access) ISP network* category.
- Figure 41: from *Applications* to the End of the automated procedure. This macro stage concerns the problems in the *public Internet and Destination* category.
- Figure 42: DNS checking stage is detailed.

4.4 Mobile cellular networks

As far as the scheme of mobile cellular networks diagnosis is concerned, its principle of functioning remains the same as for wired and wireless LANs. Nevertheless, we already pointed out the main issue of implementing for a smartphone. Compared to a computer device, resources (CPU and battery) are limited as well as some tools are not allowed on a smartphone. As we noted that this work is intended for implementing an automated system for computer device, we still included checking steps specific to computer device, which are displayed as green filled blocks. If we wanted to implement it for a smartphone, these stages would be skipped over.

Although the global structure of the sub-diagrams intended for cellular may be similar to LANs, there are still some differences that it is important to note.

- Figure 43: Start from *Settings* to *Local Link (1)*. In this macro stage, compared to LANs, we considered that IP address allocation is only in dynamical mode, in this case, it is DHCP.
- Figure 44: from *Local Link (1)* to *Local Link (2)*. In this step, Authentication and DHCP checking are only carried out on a computer device as it relies on traffic analyser. The principle is the same as for LANs.

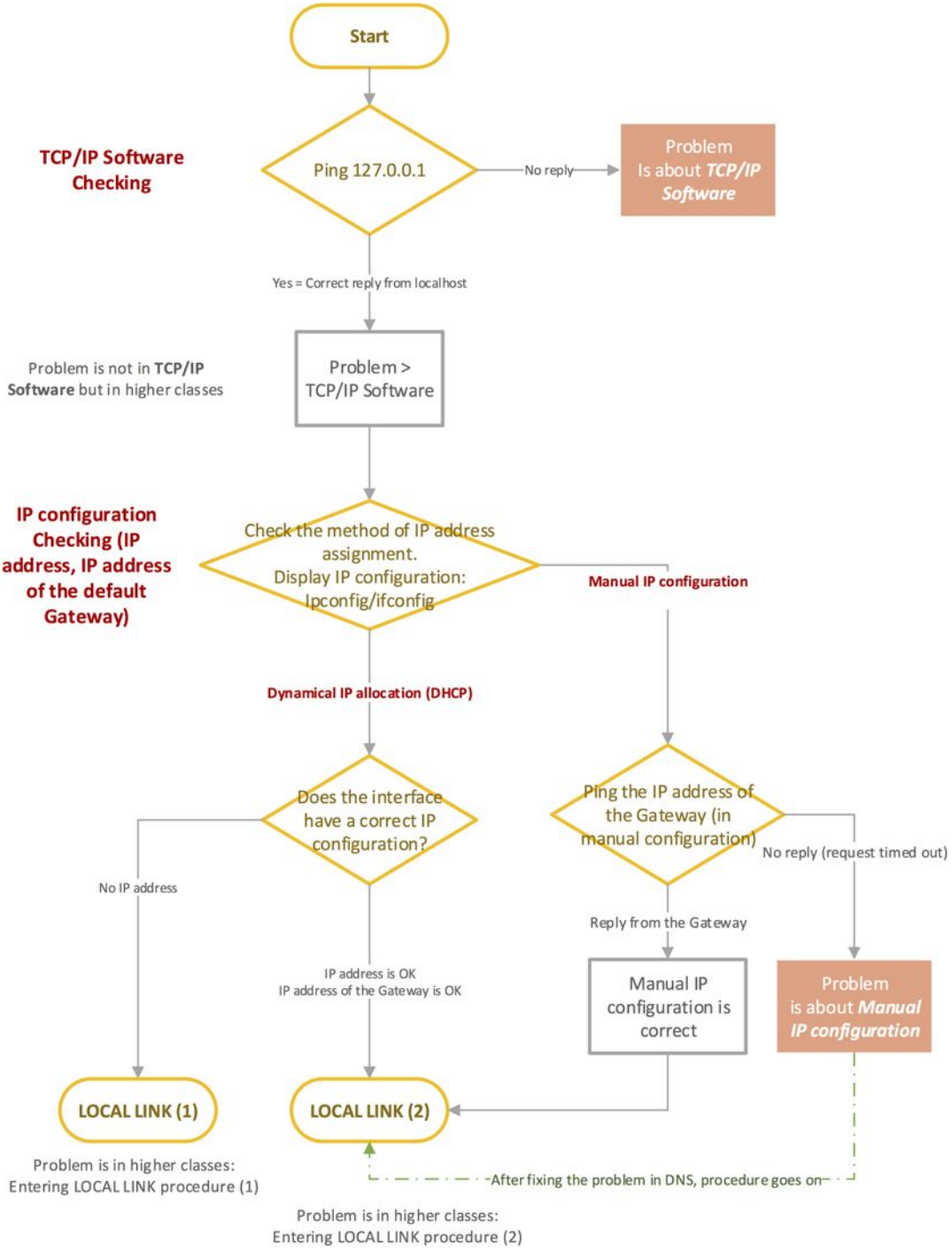


Figure 37: LAN/WLAN: The sub-diagram for Settings macro step

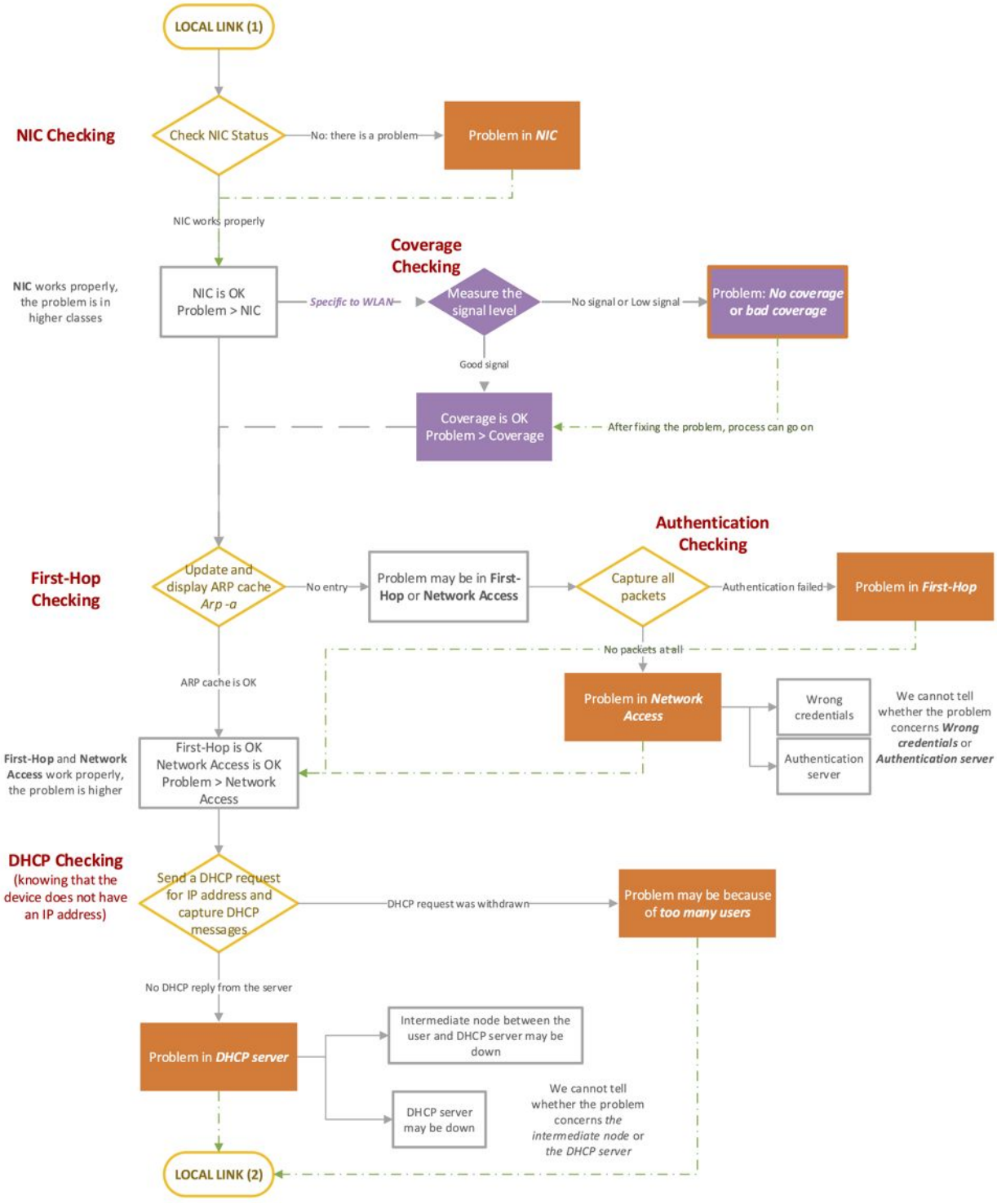


Figure 38: LAN/WLAN: The sub-diagram for Local Link (1) macro step

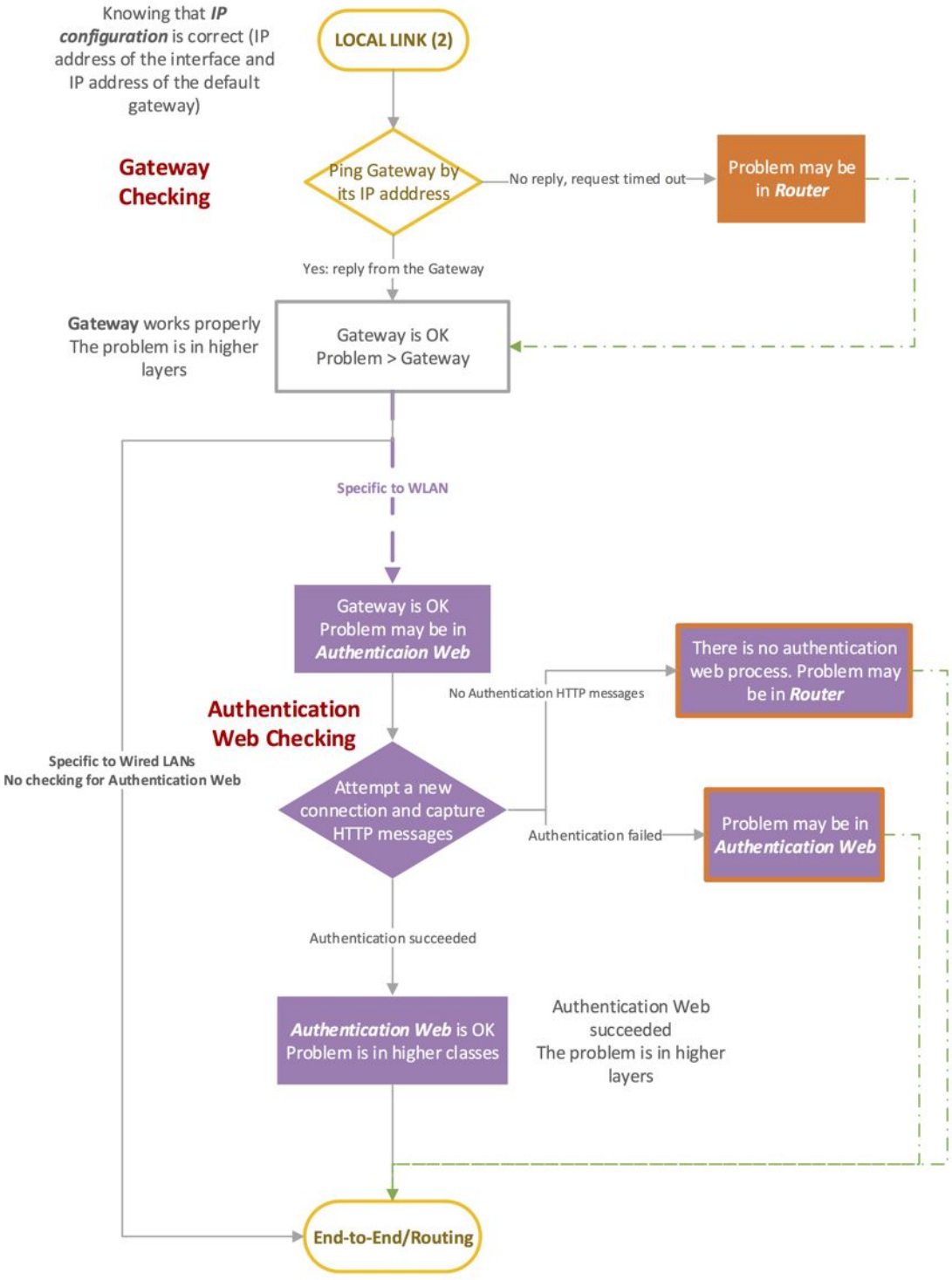


Figure 39: LAN/WLAN: The sub-diagram for Local Link (2) macro step

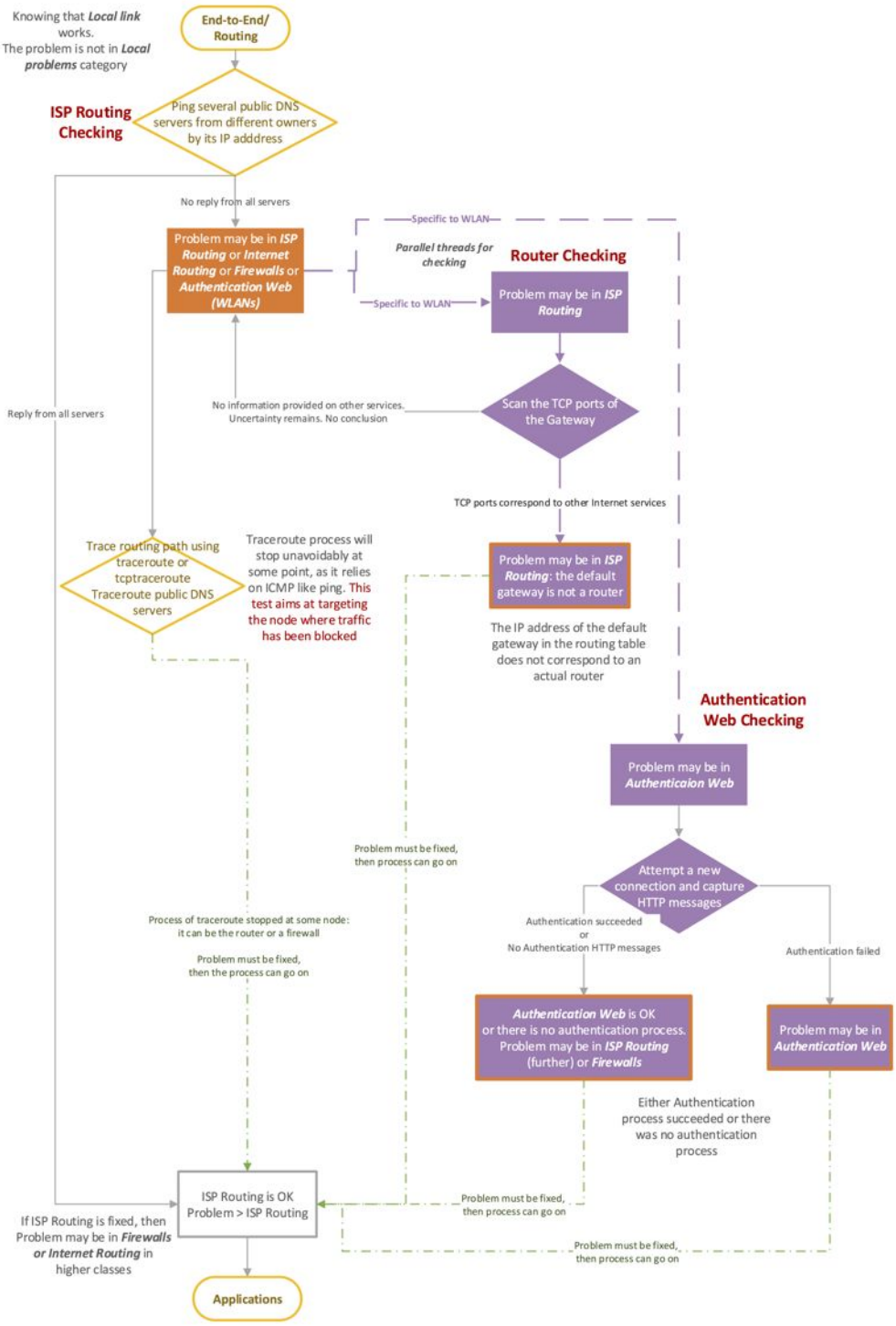


Figure 40: LAN/WLAN: The sub-diagram for End-to-End/Routing macro step

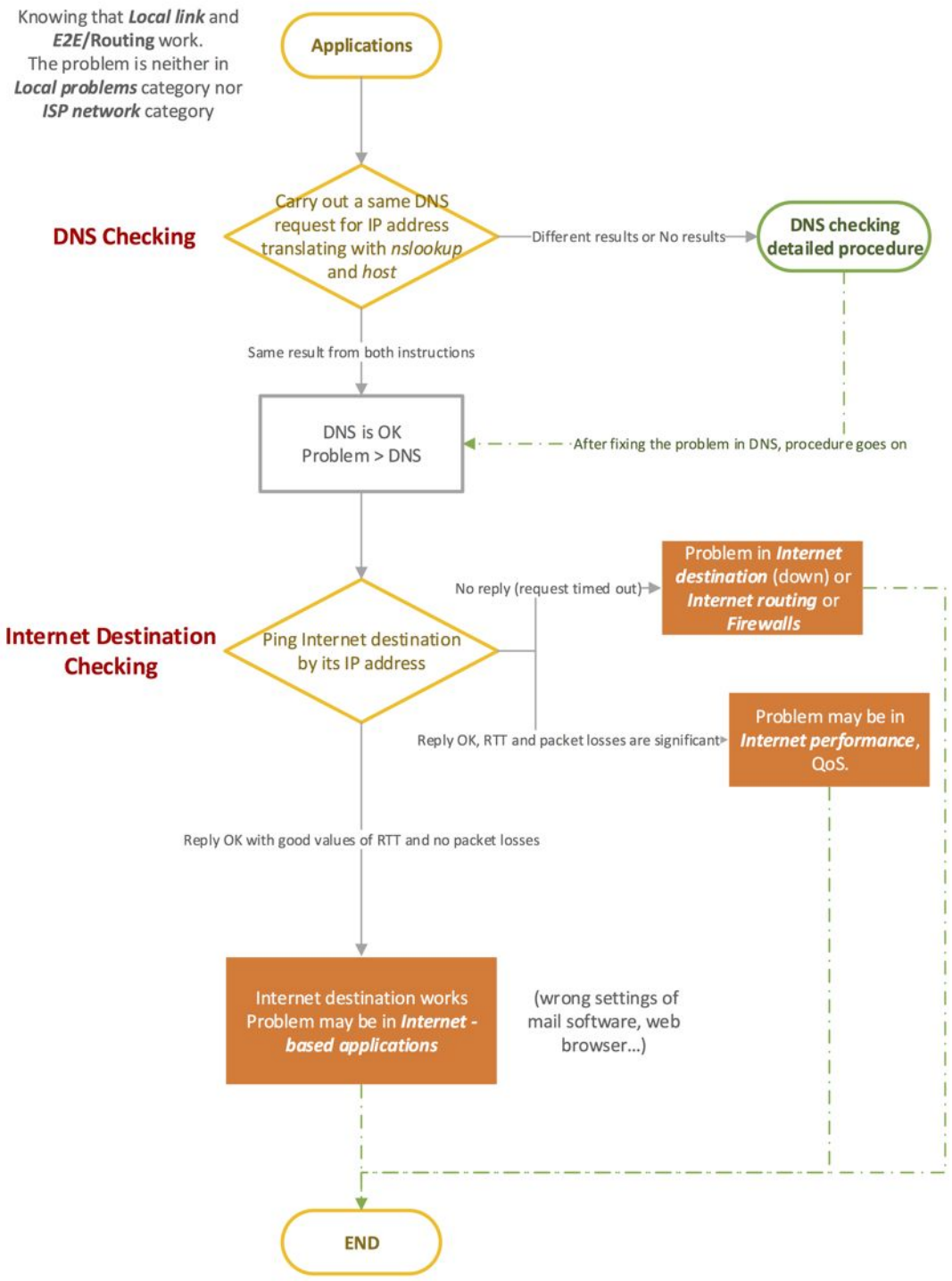


Figure 41: LAN/WLAN: The sub-diagram for Applications macro step

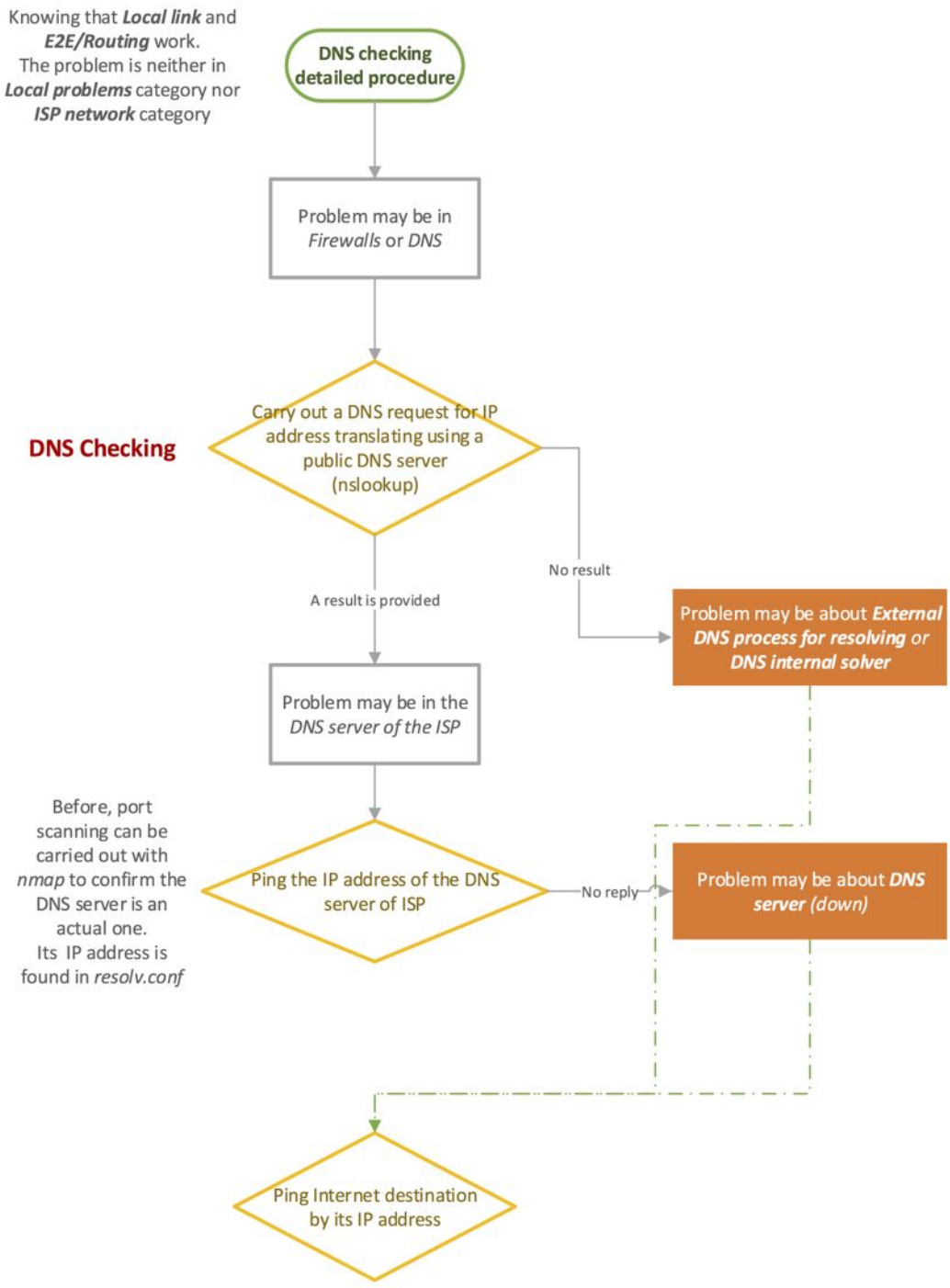


Figure 42: LAN/WLAN: The sub-diagram for DNS stage

- Figure 45: from *Local Link (2)* to *End-to-End/Routing*.
- Figure 46: from *End-to-End/Routing* to *Applications*. We decided that an uncertainty would remain regarding whether the problem is about ISP routing or Data Limit, which is actually a network access problem.
- Figure 47: from *Applications* to the End of the automated procedure. A short test in DNS stage is not performed on smartphone as it relies on tcp port scanning.

4.5 About concrete implementing

With this theoretical and diagrammatic presentation of the automated procedure for network diagnosis, there is no restriction regarding the programming language to implement such an automated system, except the device on which it would work.

Although we noted that this work focuses on the theoretical and schematic implementation of the tool, we made a few comments about which operating systems could be used, as our tests rely on applications designed in these environments: Unix-based operating systems (MAC OS, Linux, Android for smartphone) and Windows for instance.

Actually, based on the used application commands in the tests such as `ping`, `nslookup`, `host`, `nmap`, traffic analyser (`tcpdump`, `wireshark`), a simple automated system could be implemented with a BASH script [24], that would contain the suggested instructions mentioned in the tests in the specific order displayed in the state diagrams. For each instruction, we collect the output results in a temporary cache, so that it can be analysed, and depending on the result, a next instruction is carried out. As an example, the Figure 48 illustrates the comparison between instructions in a state diagram and its translating in BASH script. It displays the possible results that a `ping` command would return.

4.6 Current tools

This subsection deals with current network diagnostic tools that have been proposed to users. These software aim at helping the network administrators to manage their network and verifying the good functioning of its components.

Currently, many professional network diagnostic tools have been proposed by diverse companies, but as it is explicitly evoked, several tools are intended for professional network administrators managing substantial networks in companies, institutions, and so forth. Actually, these software tools deal more with Network management ([43, Ch4, p162-193], [29, Ch9, p798-828]) by performing Network monitoring, or bandwidth usage based on protocols such as Simple Network Management Protocol (SNMP, [15]), Packet sniffing, NetFlow ([17]), etc. For example, certain network diagnostic software like *Paessler Routing Traffic Grapher* (PRTG) from

[24] `http://www.gnu.org/software/bash/bash.html`

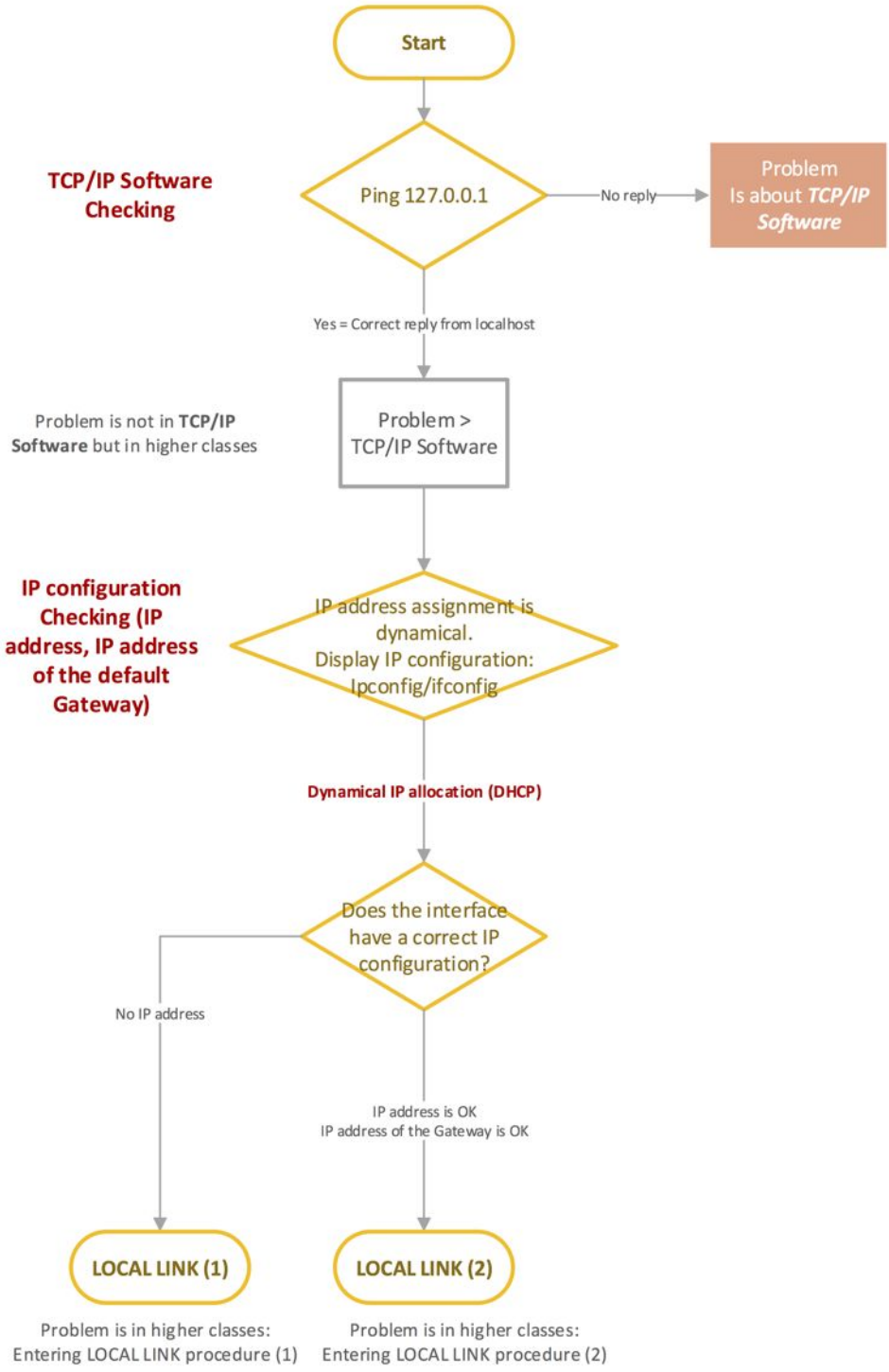


Figure 43: Cellular networks: The sub-diagram for Settings macro step

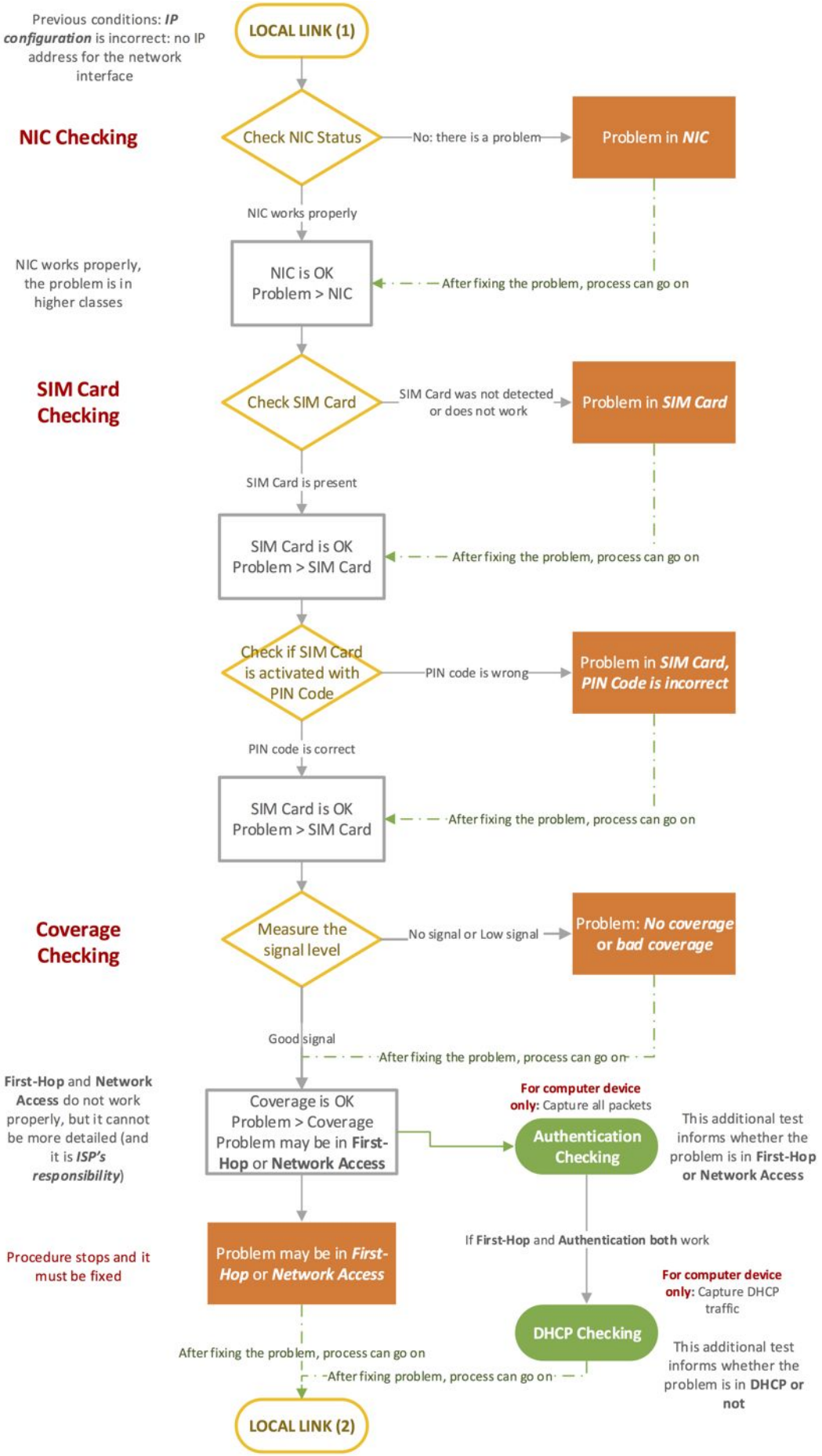


Figure 44: Cellular networks: The sub-diagram for Local Link (1) macro step

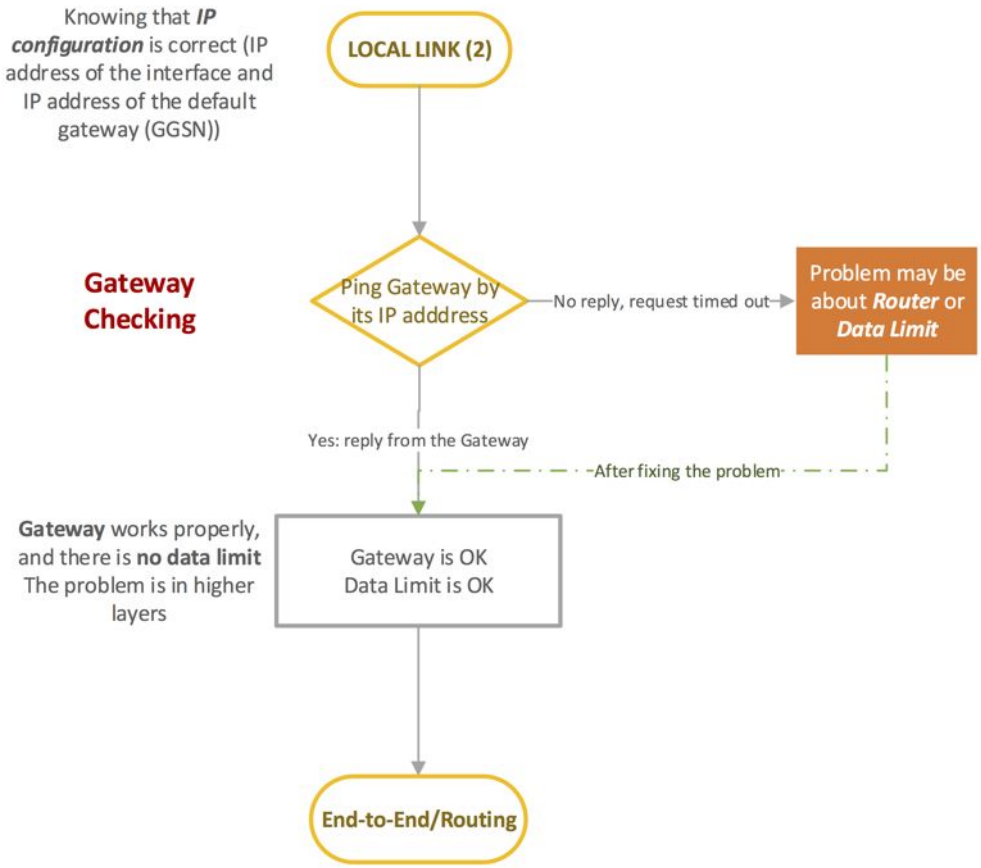


Figure 45: Cellular networks: The sub-diagram for Local Link (2) macro step

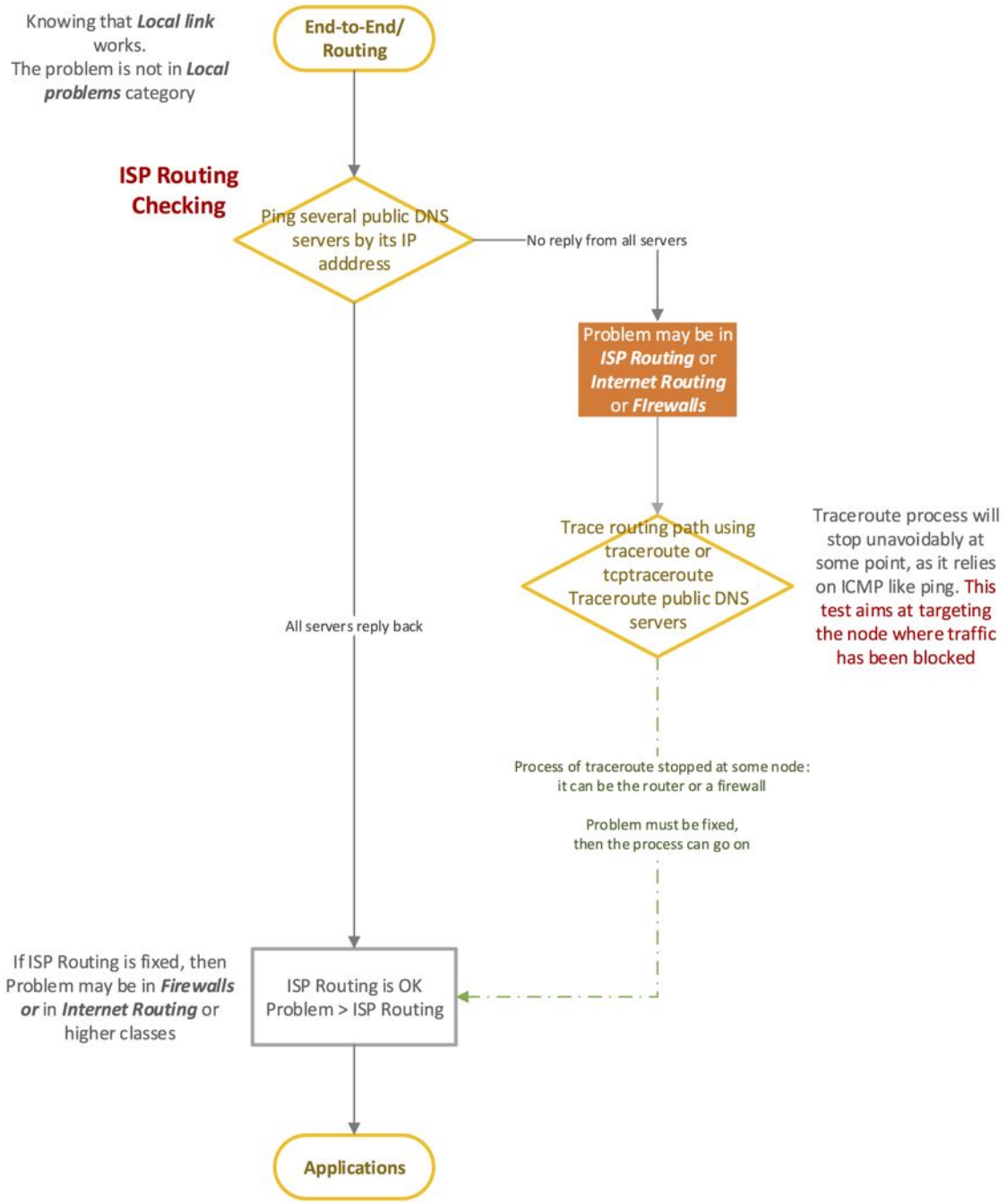


Figure 46: Cellular networks: The sub-diagram for End-to-End/Routing macro step

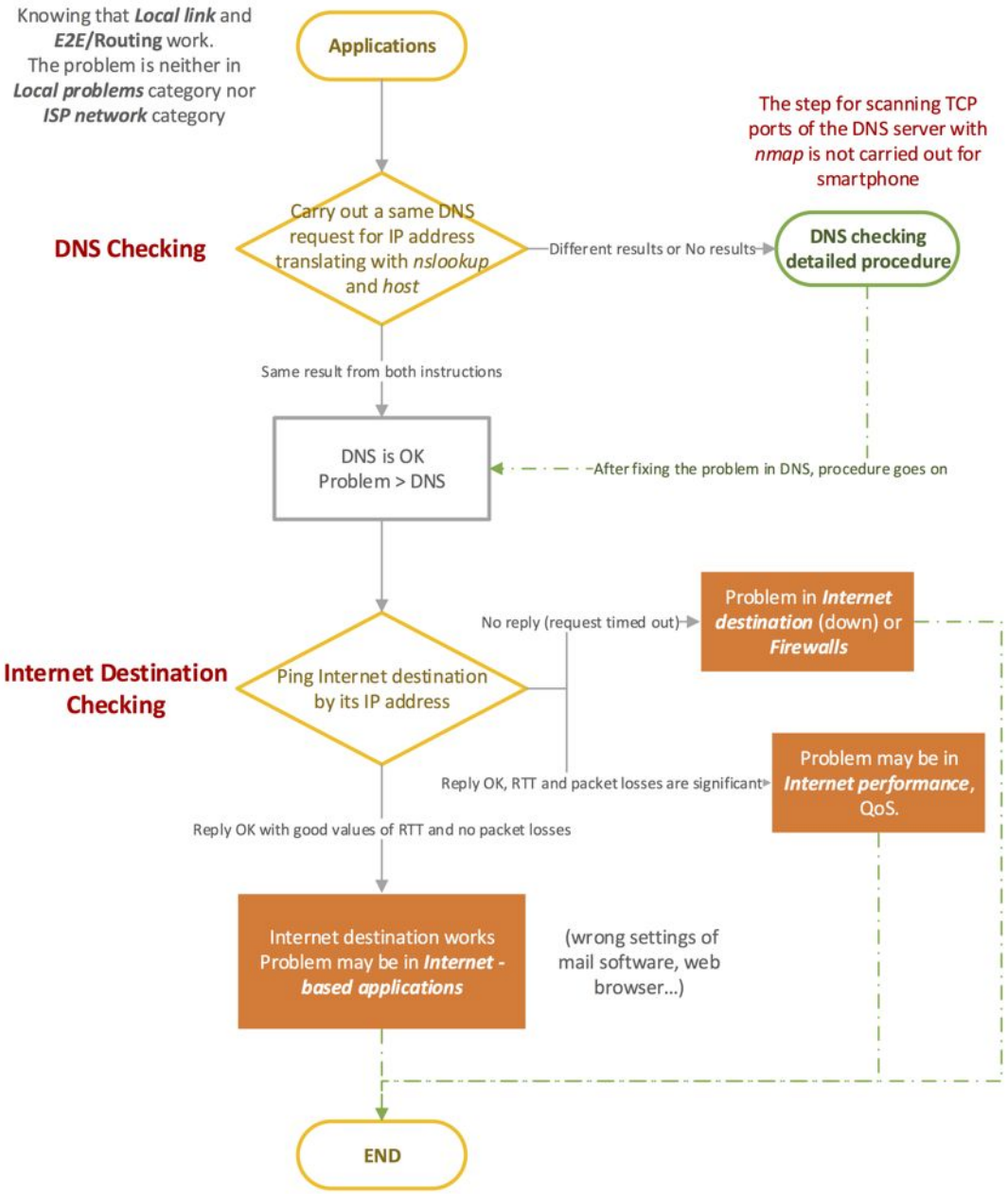


Figure 47: LAN/WLAN: The sub-diagram for Applications macro step

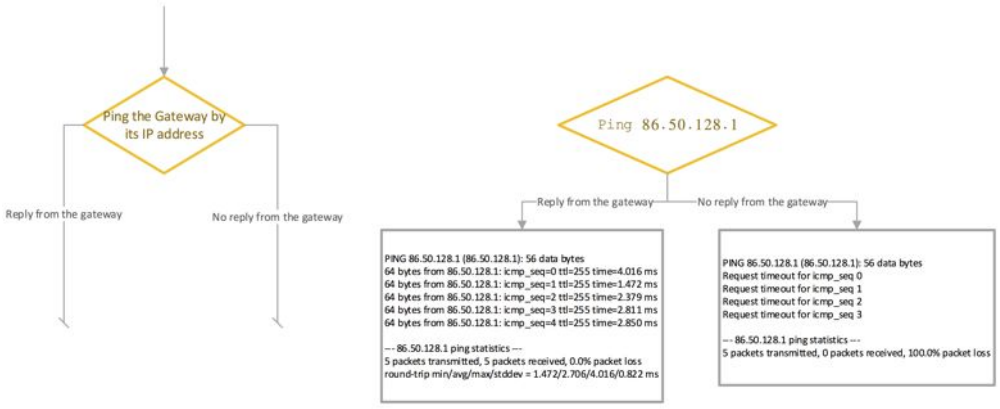


Figure 48: From the schematic state diagram to writing the BASH script

Paessler, the network monitoring company [25] offers this type of services. It helps to determine the status of the routers, the firewalls and the other components of the network (Figure 49, available at [26]), to display real-time throughput traffic and resources of the devices, and it also sends alarms and notifications about instructions which failed like pinging a host for instance. Other tools from companies such as Netvision [27], Riverbed [28] offer tools intended for network monitoring and also Internet performance management, which is also a class of problem mentioned in the classification of problems. Free software that provide identical services are also available like Observium [29]. However, the main issue of the previously evoked tools is the fact that it may be too complicated for simple users who just need to browse on the World Wide Web. Indeed, these tools are intended for professional network administrators who do have significant knowledges of functioning of the Internet.

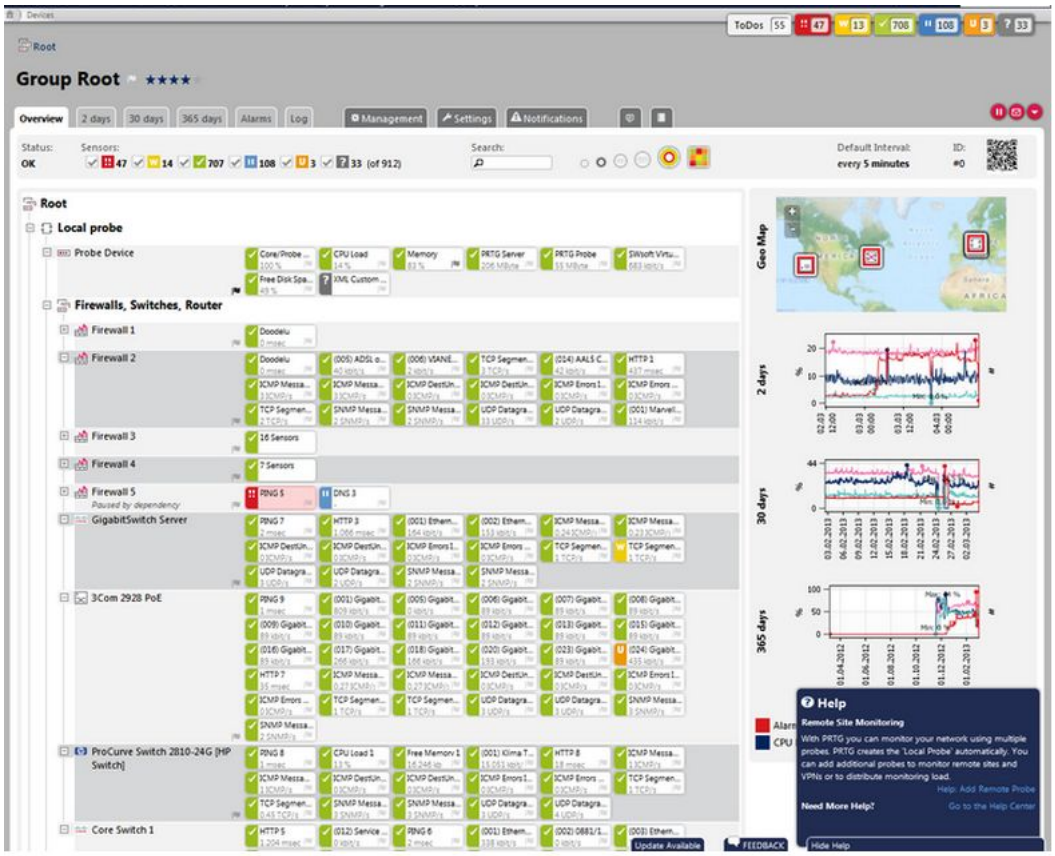


Figure 49: Screenshot of an overview PRTG network monitoring

As far as open source software are concerned, the research community Internet2 [30] works on proposing useful tools for network and multimedia. For instance, Network

[25] http://www.paessler.com/prtg
[26] http://www.paessler.com/prtg/screenshots
[27] http://www.netvision.com/
[28] http://www.riverbed.com/
[29] http://www.observium.org/wiki/Main_Page
[30] http://www.internet2.edu/

Diagnostic Tool, i.e NDT [31] provides network configuration and performance testing, but it is actually more like an advanced Internet "speed test", with additional data about throughput analysis, buffer sizes, RTTs, loss rate, etc. Once again, this tool may not be really user-friendly for users that are not familiar with Linux operating systems, especially as the user will not be familiar with the detailed data and may not be interested in.

Of course, we can mention the standard network tools to ping, perform dns resolving, port scanning, i.e `ping`, `traceroute`, `nslookup`, `host`, `nmap`, etc. The same issue is that users may not be familiar with these UNIX-based tools, that is why developers proposed web-applications based ton the principle of these tests, like Network Tools [32] that carries out some of the previous network tools, DNSStuff [33] for DNS resolving, etc. It is relevant to note that these tools are specific to a particular test for a specific class, and that it relies on the fact that the basic connectivity already works, i.e the traffic goes through the local network and are routed to the host performing this web-based applications.

Regarding Network diagnosis tools included in the operating system, it is noted that it is not specifically intended for network diagnosis. The main problem of the Help section for network troubles in Windows is that it is not an automated process. The user is suggested to perform a set of some short tests like what we proposed in Chapter 3 as shown in Windows Help section at [34]. Windows Help displays a list of possible problems that the user must understand before carrying out possible solutions to fix the problem. The issue of such a table is that the user must identify the problem at first. For instance, the table of problems on Windows Help section expounds different possible problems such as "Can't connect to a home network (wired connection)" and proposes different solutions to fix it like "Make sure the router and modem are turned on". A simple user that is not familiar with network technology cannot even tell if he cannot connect to a home network, because its only tool to check it is usually the web browser, that relies on several lower protocols. However, the fact that a web browser cannot fetch a webpage from a website does not imply that the problem is about the connection to a home network. It may be in DNS resolving for instance.

Although the tests suggested by Windows are correct, the user may not have enough knowledges to choose the correct test to perform, and may not have an idea about how to perform it neither. Even if assuming that the user can identify the problem such as "Can't connect to a home network (wired connection)" in Windows Help section, he is suggested to "make sure the Ethernet cable is connected to the correct port on the router" for instance. Although more information are detailed about it:

> The Ethernet cable from your computer to the router should not be connected to a port marked "uplink," "WAN," or "Internet." Some

[31] `http://www.internet2.edu/performance/ndt/index.html`
[32] `http://network-tools.com/`
[33] `http://www.dnsstuff.com/tools`
[34] `http://windows.microsoft.com/en-gb/windows-vista/troubleshoot-network-connection-problems`

routers disable the port next to the uplink port, so try using a different one.

. The user who had never worked on this problem may be lost.

To sum up, there is currently a wide range of tools and software for network troubleshooting, targeting a diverse range of users, but most of the tools we mentioned in this part are intended for users that already have a sufficient background about the Internet technologies, and are also specific to classes such as measuring the Internet performance, checking the DNS services, etc. A simple user that is not comfortable with these technologies may be lost with such tools. He needs a tool that would complete a whole automated checking that combines all the specific tests for all classes while being not to complex regarding the information provided. Indeed, the simple user especially wants to know who is to be held liable for the fixing the problem, and who is not, that is to say either himself or the access ISP network or else the other ISPs in the Internet.

In this chapter, we presented the schematic procedure with detailed state diagrams. Admittedly, these state diagrams are only theoretical schemes, but it is a necessary stage before implementing such a tool for real. We also proposed a short study about the kind of current tools for network diagnosis tools that are proposed. In the next chapter, we will review our study and discuss about improvement and additional services for our automated system.

5 Discussion and conclusions

The main objective of our work was to implement a schematic algorithm in order to process a network diagnosis in the event that there is a malfunction in Internet connectivity or in Internet-based applications. The essential keyword of the problematic is the characteristic of the algorithm that should be performed in a form of an *automated* procedure to make the network troubleshooting easier for the users. Indeed, we already pointed out the fact that due to a lack of knowledges about the Internet technologies, a standard user is usually lost face to help sections about manual network troubleshooting provided in some operating systems.

5.1 A theoretical design

In order to design such an automated algorithm for network troubleshooting, we tried to take into account as many classes of problems as possible that the user may encounter while using Internet services. We classified these causes of trouble, based on the TCP/IP architecture, and we proposed a test to target each one the classified classes. Each test is actually a combination of shorter tests, executed in a certain order. These ordered combinations were determined based on analysing the results and information each shorter test could provide.

For our automated system, we considered three network technologies intended for Internet access, which are Ethernet technology for wired LANs, IEEE 802.11 for wireless networks and GPRS/UMTS for mobile cellular networks. Admittedly, our scheme is still global, and we presented a global structure regarding the layers of the TCP/IP architecture, which could be suitable for OSI model. Consequently, our automated system could be adapted to other network technologies as their architecture relies on OSI model. For instance, the system could be adapted to IEEE 802.16 (WiMAX), and also new generations for mobile cellular networks like 4G LTE.

As a matter of fact, it is essential to propose a global schema that can be adapted to a wide range of network technologies, as IT technologies and infrastructure evolve because of the increasing requirement to be real-time connected.

5.2 Tests results and limitations

Our work aimed at developing a schematic algorithm meant for later concrete implementation. Although we suggested some already existent application commands in certain tests, we tried to define more the *principle of functioning of each test*, that is to say, what must be verified by a checking process, which type of result it should return to the user. It is not mandatory to use the suggested application commands explained in the tests, like `ping` for instance. The developer is free to choose a different way to implement this test, but it must respect the principle of the test which is, in this very case, to confirm that the requested host is up and replies back, implying that it can provide specific services.

The main issue of our theoretical design of the automated procedure is that it is actually only theoretical. In most of situations of Internet problems we discussed

about, we could not verify for real that the combination of short tests we suggested could work. Indeed, except the classes of problems identified in *Local problems* category, the other causes of trouble were external to the local access network and more uncommon, for instance DNS problems (resolving does not work). Obviously, as a simple user, we did not have access to the management of an ISP network to generate problems in order to confirm that our algorithm works. Nevertheless, in certain situations, we tried to check our short tests in order to determine the correct ordered combination of short tests. For instance, this was the case for the situation in which *Authentication Web* class was the origin of the non-access to Internet applications, i.e the World Wide Web in this case.

Moreover, we noted that we tried to identify and classify as many classes of problems as possible, but not all of the mentioned classes could have been targeted separately such as *Firewalls*, and *Internet performance*. In some stages of the algorithm, there is a doubt about at least two possible classes of problems. Actually, *Firewalls* problems are not easy to target when it is out of the local access network. For a firewall at the border between the local access network and the ISP network, an attempt could have been to generate different types of traffic (TCP ports, protocols) and observe the packet capturing. *Internet performance* is a complexed class of problem that should be considered as a whole independent analysis because many parameters and factors are involved. However, in the point of view of a simple user, the problem of the Internet performance represents two simple options, more estimated in the way of Mean Option Score: the Internet connection is bad or good.

Besides, we encountered other limitations to implement our procedure. We assumed in our work that this algorithm was intended for a "standard Internet" connection. We defined this type of Internet connection as common situations where a simple user wants to connect his computer/laptop to a local home network (wired or wireless), or a student trying to get access to the university wireless network he is not responsible for, and finally a mobile user who wishes to access to the Internet through cellular network from his smartphone. However, there are other methods and technologies to connect to a host via the Internet. Indeed, in this work, we did not consider specific network connections such as Virtual Private Networks [35], connections based on Secure Shell ([49]), or also Internet access from virtual machines for instance. Another interesting case would have been to design the automated procedure to intend it for multi-homing devices, i.e, devices that are connected to more than one network via different network interfaces.

5.3 Implementing for real use and future work

As we recall that our algorithm is theoretical, all the tests that have been suggested may not be feasible depending on the device it is intended for. For instance, it could be due to policy of usage, like packet capturing. Actually, the essential characteristic of the tests are more the network protocols it is based on than the already existent applications to perform it. We tried to provide detailed explanations about the

[35] `http://technet.microsoft.com/en-us/library/bb742566.aspx`

network protocols the tests rely on, more than the application command in order to adapt it for different devices and situations. For example, `ping` application could be adapted to smartphone for instance, but the principle of using the ICMP protocol in the tests must remain.

It is important to respect the automated characteristic of the automated-system intended for users. The automated-system must meet the users needs, that is to say, to provide clear information without being too exhaustive. Indeed, the user is actually a "user", which means that he may be interested only in using the service as long as it works. For instance, informing with too many details that the Internet connection does not work because the external DNS resolving procedure failed would not interest the user, because the main fact is that the Internet connection does not work, and it is the ISP that is responsible for that. Consequently, the amount of information provided must be adapted to the type of users the automated tool is intended for.

Trying to solve the problem To discuss further, after implementing such an automated-system, some improvements could be added, like another automated procedure that will be carried out after the network diagnosis in order to fix the problem once it has been identified. Indeed, in the event that it is possible to fix it, the user may like to be guided easily with automated instructions. Another procedure could be implemented to do so and help the user to fix the Internet connection. This automated procedure should be considered as a whole issue.

Actually, referring to the three main environments we defined to make the classification of problems in the Internet, i.e the *Local problems*, the *access ISP network* and the *public Internet with the destination host*, it can be noted that only some problems in the *Local problems* category may be solved by the user himself. Indeed, the problems in *ISP network* and *Destination* concern other institutions. Nevertheless, *Local problems* can be fixed in a particular situation which is when the user configured himself his own local network. For instance, it could be a home network connecting many devices such as computers, laptops, printers, and so forth. It implies that the user may have configured a router, maybe switches, a DHCP server to allocate IP addresses, and possibly a DNS server although it is usually provided by the access ISP. Another example is a more complexed network in a business company which manages a substantial number of devices. In a nutshell, the issue of fixing the problems involves that the user is the network administrator of its local network. This is not the case for a student connecting his device to the University network, or a user getting Internet access on his smartphone via cellular mobile networks. Admittedly, this case corresponds more to the case of configuring its own home network for instance, where the user may need to configure a router and switches.

Repairing is a whole issue, because it requires specific and diverse technical skills like configuring routers, DHCP servers, DNS servers, switches, and of course the device itself, including the network adapters, the TCP/IP software and the Internet-based applications. Usually, a standard user that does not have a sufficient background in the Internet functioning cannot do more than resetting his device

and the software (TCP/IP software and Internet-based applications) if there is a problem.

Although the user may not be able to fix the problem, the main point is that he is able to locate it. It is essential because it provides details about the correct institution to be actually blamed for or not. Moreover, trying to solve the problem at all costs may end in using illegal means like for instance trying to get round firewalls by connecting to a proxy server so that it can perform the requests instead, or even using malicious methods like performing IP spoofing.

References

[1] http://media.popularmechanics.com/images/smart-house-430.jpg. retrieved in February 2013.

[2] http://public-dns.tk/, retrieved in September 2013.

[3] inssider for home. http://www.metageek.net/products/inssider/.

[4] Ethernet based lans. IEEE Standard 802.3, Institute of Electrical and Electronics Engineers, 1983-2010. Archive available at http://www.ieee802.org/3/.

[5] Wireless local access networks. IEEE Standard 802.11, Institute of Electrical and Electronics Engineers, 1997-2012. archive availble at http://www.ieee802.org/11/.

[6] 3g. International Mobile Telecommunications 2000, International Telecommunication Union, 2001-2003. Available at http://www.itu.int/osg/imt-project/docs/What_is_IMT2000.ppt, retrieved in June 2013.

[7] How to use web-authentication. Technical report, Allied Telesis, August 2010. Available at http://www.alliedtelesis.com/media/fount/how_to_note_alliedware_plus/howto_aw_plus__use_web_authentication.pdf., retrieved in July 2013.

[8] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz. Extensible authentication protocol (eap). Request For Comments 3748, Internet Engineering Task Force, June 2004. Also available at http://tools.ietf.org/html/rfc3748.

[9] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform resource identifier (uri): Generic syntax. Request For Comments 3986, Internet Engineering Task Force, January 2005. Also available at http://tools.ietf.org/html/rfc3986.

[10] John Gilmore Bill Croft. Bootstrap protocol (bootp). Request For Comments 951, Internet Engineering Task Force, September 1985. Also available at http://tools.ietf.org/html/rfc951.

[11] R. Braden. Requirements for internet hosts – application and support. Request For Comments 1123, Internet Engineering Task Force, 1989.

[12] R. Braden. Requirements for internet hosts – communication layer. Request For Comments 1122, Internet Engineering Task Force, October 1989. Also available at http://tools.ietf.org/html/rfc1122.

[13] R. Braden. Information technology - open systems interconnection - basic reference model: The basic model. International Standard 7498-1, International Organization for Standardization and International Electrotechnical Commission, 1994.

[14] John F. Buford, Heather Yu, and Eng Keong Lua. *P2P Networking and Applications*. Morgan Kaufmann, 2009.

[15] J. Case, M. Fedor, M. Schoffstall, and J. Davin. A simple network management protocol (snmp). Request For Comments 1157, Internet Engineering Task Force, May 1990. Also available at `http://tools.ietf.org/html/rfc1157`.

[16] S. Cheshire, B. Aboba, and E. Guttman. Dynamic configuration of ipv4 link-local addresses. Request For Comments 3927, Internet Engineering Task Force, May 2005.

[17] B. Claise. Cisco systems netflow services export version 9. Request For Comments 3954, Internet Engineering Task Force, October 2004. Also available at `http://www.ietf.org/rfc/rfc3954.txt`.

[18] R. Droms. Dynamic host configuration protocol. Request For Comments 2131, Internet Engineering Task Force, March 1997. Also available at `http://tools.ietf.org/html/rfc2131`.

[19] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext transfer protocol – http/1.1. Request For Comments 2616, Internet Engineering Task Force, June 1999. Also available at `http://www.ietf.org/rfc/rfc2616.txt`.

[20] Helsinki: Statistics Finland. Internet users, spring 2001 to spring 2007, percentage of 15 to 74-year-olds by age group. Available at `http://www.stat.fi/til/sutivi/2007/sutivi_2007_2007-09-28_kuv_004_en.html.`, retrieved in February 2013, 2007. Use of information and communications technology by individuals.

[21] Helsinki: Statistics Finland. Places of internet use in spring 2007, per cent of 15-74 years old population. Available at `http://www.stat.fi/til/sutivi/2007/sutivi_2007_2007-09-28_kuv_011_en.html.`, retrieved in February 2013, 2007. Use of information and communications technology by individuals.

[22] Helsinki: Statistics Finland. Purposes of the use of the internet in spring 2008, per cent of internet users by age group. Available at `http://www.stat.fi/til/sutivi/2008/sutivi_2008_2008-08-25_tau_001_en.html`, retrieved in February 2013, 2008. Use of information and communications technology by individuals.

[23] Helsinki: Statistics Finland. Prevalence of internet usage and certain purposes of use in 2012. Available at `http://www.stat.fi/til/sutivi/2012/sutivi_2012_2012-11-07_tie_001_en.html`, retrieved in February 2013, 2012. Use of information and communications technology by individuals.

[24] Robert Graham. Apple's secret "wispr" request. `http://blog.erratasec.com/2010/09/apples-secret-wispr-request.html#.UkSf0GROpbX`.

[25] Adrian Granados. Scan, find and troubleshoot wireless networks with wifi explorer. http://www.adriangranados.com, 2013.

[26] Gunnar Heine and Holger Sagkob. *Gprs: Gateway to Third Generation Mobile Networks*. Artech House, January 2003.

[27] IANA. Service name and transport protocol port number registry. http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml, September 2013.

[28] Alban Jacquemin and Adrien Mercier. Les firewalls. `http://www.frameip.com/firewall/`, retrieved in September 2013, 2004-2005.

[29] Keith W. Ross James F.Kurose. *Computer Networking, a Top-Down Approach*. Pearson International Edition, 2010.

[30] Tony Jeffree. Port based network access control. IEEE Standard 2004, Institute of Electrical and Electronics Engineers, 2003-2004. Available at `http://www.ieee802.org/1/pages/802.1x-2004.html`, retrieved in September 2013.

[31] Amos E. Joel. Asynchronous transfer mode. Ieee press, Institute of Electrical and Electronics Engineers, 1993.

[32] Juha Korhonen. *Introduction to 3G Mobile Communications*. Artech House, 2003.

[33] Charles M. Kozierok. The tcp/ip guide. available at `http://www.tcpipguide.com/free/t_DHCPGeneralOperationandClientFiniteStateMachine.htm`, retrieved in April 2013, 2003-2012.

[34] P. Mockapetris. Domain names - concept and facilities. Request For Comments 1034, Internet Engineering Task Force, November 1987. Also available at `http://tools.ietf.org/html/rfc1034`.

[35] P. Mockapetris. Domain names - implementation and specification. Request For Comments 1035, Internet Engineering Task Force, November 1987. Also available at `http://tools.ietf.org/html/rfc1035`.

[36] Mike Muuss. Ping - linux man page. Manual page available at `http://linux.die.net/man/8/ping`, retrieved in September 2013, December 1983.

[37] Jaime Peñalba. Netdiscover. http://nixgeneration.com/ jaime/netdiscover/.

[38] Guillaume Plouin. *Sécurité, stratégie d'entreprise et panorama du marché, 3ème édition*. InfoPro, Dunod, 2013.

[39] David C. Plummer. An ethernet address resolution protocol. Request For Comments 826, Internet Engineering Task Force, November 1982. Also available at `http://tools.ietf.org/html/rfc826`.

[40] J. Postel. User datagram protocol. Request For Comments 768, Internet Engineering Task Force, August 1980. Also available at `http://tools.ietf.org/html/rfc768`.

[41] J. Postel. Internet control message protocol. Request For Comments 792, Internet Engineering Task Force, September 1981. Also available at `http://tools.ietf.org/html/rfc792`.

[42] J. Postel. Transmission control protocol. Request For Comments 793, Internet Engineering Task Force, September 1981. Also available at `http://tools.ietf.org/html/rfc793`.

[43] Julia L. Panko Raymond R. Panko. *Business Data Networks and Telecommunications, 8th Edition*. Pearson International Edition, 2011.

[44] Siegmund M. Redl, Mathias K. Weber, and Malcolm W. Oliphant. *An Introduction to GSM*. Artech House, 1995.

[45] C. Rigney, A. Rubens, W. Simpson, and S. Willens. Remote authentication dial in user service (radius). Request For Comments 2058, Internet Engineering Task Force, January 1997. Also available at `http://tools.ietf.org/html/rfc2058`.

[46] R. Shirey. Internet security glossary, version 2. Request For Comments 4949, Internet Engineering Task Force, August 2007. Also available at `http://tools.ietf.org/html/rfc4949`.

[47] C. Kale T. Socolofsky. A tcp/ip tutorial. Request For Comments 1180, Internet Engineering Task Force, January 1991. Also available at `http://tools.ietf.org/html/rfc1180`.

[48] Thomas Case William Stallings. *Business Data Communications, Infrastructure, Networking and Security, 7th edition*. Pearson International Edition, 2013.

[49] T. Ylonen and C. Lonvick. The secure shell (ssh) protocol architecture. Request For Comments 4251, Internet Engineering Task Force, January 2006. Also available at `http://tools.ietf.org/html/rfc4251`.